Unsupervised Texture Segmentation Using Multiresolution Markov Random Fields

by

Chang-Tsun Li

Submitted for the degree of Doctor of Philosophy
to the Higher Degree Committee

Department of Computer Science
University of Warwick
Coventry CV4 7AL, UK
ctli@dcs.warwick.ac.uk

Unsupervised Texture Segmentation Using Multiresolution Markov Random Fields

Chang-Tsun Li

Summary

In this thesis, a multiresolution Markov Random Field (MMRF) model for segmenting textured images without supervision is proposed. Stochastic relaxation labelling is adopted to assign the class label with highest probability to the block (site) being visited. Class information is propagated from low spatial resolution to high spatial resolution, via appropriate modifications to the interaction energies defining the field, to minimise class-position uncertainty.

The thesis contains novel ideas presented in Chapter 4 and 5, respectively. In Chapter 4, the Multiresolution Fourier Transform (MFT) is used to provide a set of spatially localised texture descriptors, which are based on a two-component model of texture, in which one component is a deformation, representing the structural or deterministic elements and the other is a stochastic one. Experiments show that the algorithm is efficient in alleviating class-position uncertainty via data propagation across resolutions. However, the blocking artifacts of the segmentation results show that it is preferable to combine both class and position information so as to achieve smoother and more accurate boundary estimation.

In Chapter 5, based on the same MFT-MMRF framework, a boundary process is proposed to refine the segmentation result of the region process proposed in Chapter 4. At each resolution, all the image blocks on either sides of the preliminary boundary detected in the region process are treated as potential boundary-containing blocks (PBCB's). The orientation and the centroid of the boundary-segment contained in each PBCB are calculated. The sequence of PBCB's are then modelled as a MRF and the interaction energy between each pair of neighbouring blocks is defined as a function of the 'distance' D between the centroids of the two boundary segments. During the stochastic relaxation process *boundary/non-boundary* labels are assigned to the PBCB's. Once the algorithm converges, the centroids of the identified true boundary blocks are connected to form the refined boundary which is propagated down to the next resolution for further refinement.

Keywords

Markov random fields, texture segmentation, deformable templates, Multiresolution

Contents

List of Tables

List of Figures

Acknowledgements

This work was conducted in the Image and Signal Processing Laboratory in the Department of Computer Science at Warwick University with the support of the Government of the Republic of China on Taiwan.

I would like to thank all the staff of the department for their administrative and technical support. I would also like to express my appreciation to Dr Kung-Hao Liang of Academia Sinica on Taiwan for the stimulating discussions and to all my colleagues and dear friends in the laboratory: James Beacom, Kuo-Huei Chen, Simon Clippingdale, Nicola Cross, Andrew King, Ian Levy, Peter Meulemans, Hugh Scott and Timothy Shuttleworth for their valuable discussions and system assistances.

I am particularly indebted to my supervisor Dr. Roland Wilson for his guidance throughout the course of this work. Without his expertise and enthusiasm this work would not have been possible.

Chapter 1

Introduction

Texture is an important and ubiquitous characteristic for the analysis of different types of image. It is present in the images of outdoor scenes and object surfaces which are frequently analysed in the fields of computer vision.. It can also be seen in multi-spectral images taken from satellites or aircraft which are analysed in the remote sensing community. In biomedicine, magnetic resonance imaging, ultrasound images of the human body and microscopic images of cell cultures or tissue samples often consist of regions of different textures [115].

1.1 What Is Texture?

Texture originally referred to the appearance of woven fabrics. Some definitions are given in Collins English Dictionary [1] as:

- "The surface of a material, esp. as perceived by the sense of touch.
- The structure, appearance, and feel of a woven fabric.
- The general structure and disposition of the constituent parts of something.

- The distinctive character or quality of something.

- The nature of a surface other than smooth."

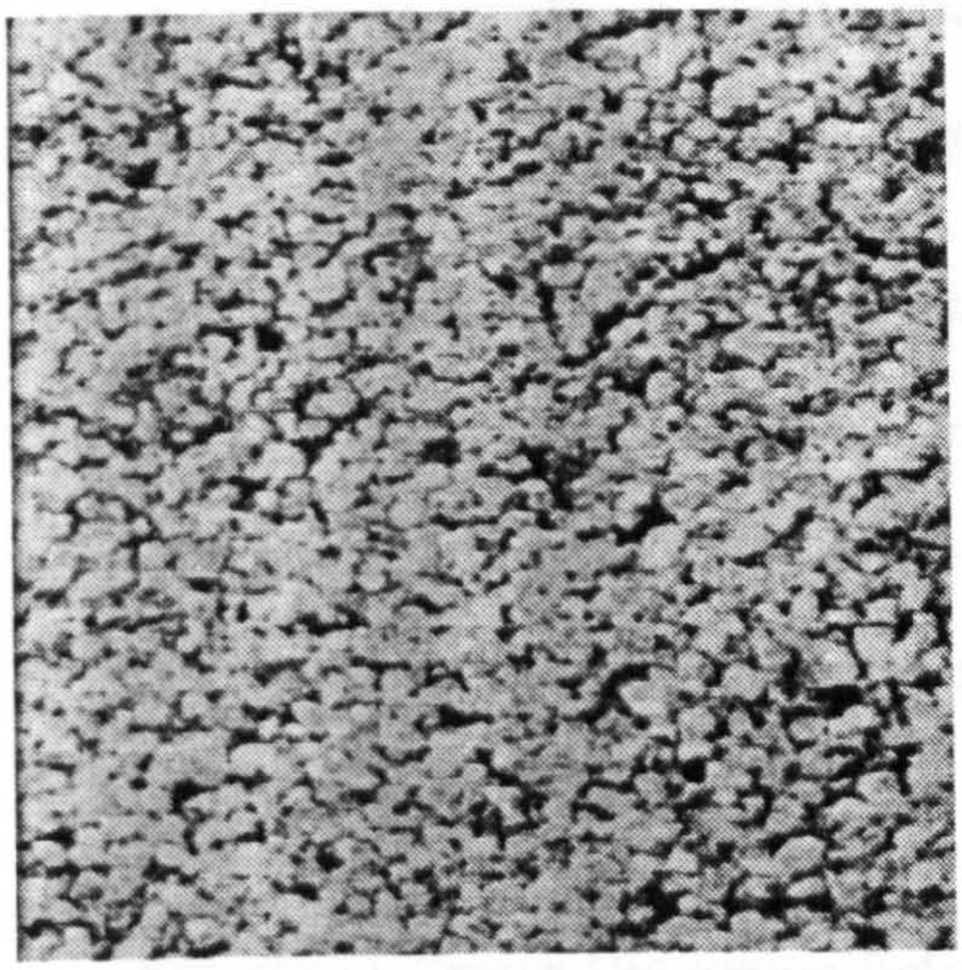
However, the definitions given above are too restrictive to describe visual textures and unfortunately, in the community of image processing or pattern recognition, no precise definition seems to exist. In the disciplines of image processing, pattern recognition and computer vision, texture is a term used to describe the properties of the surface of a given object or region encountered in an image. Informally, a texture can be described as a spatially organised area composed of a significant number of more or less regularly arranged primitives which resemble each other and give rise to the perception of regional homogeneity. Each primitive is a maximally connected set of pixels having, first, a specific gray level property such as the maximum, minimum, or the average gray level of the set, and secondly, a regional property such as area, shape, or elongation of shapes. The simplest primitive is a single pixel with its gray level. An example of a primitive with more than one pixel is a maximally connected set of pixels having the same gray level. These primitives are sometimes called texels [42]. Moreover, if levels of abstraction are taken into consideration, it is possible to find texture or sub-primitives within the primitives themselves. This non-homogeneity is called micro-texture [42] and the pattern composed of larger primitives is referred to as macro-texture [40]. Thus, texture is a regional property rather than a pixel property. It is because of this regional property that contextual constraints must be taken into account when texture is dealt with.

To characterise texture, both the local gray level properties of the primitives and the spatial relationships or interaction between them must be

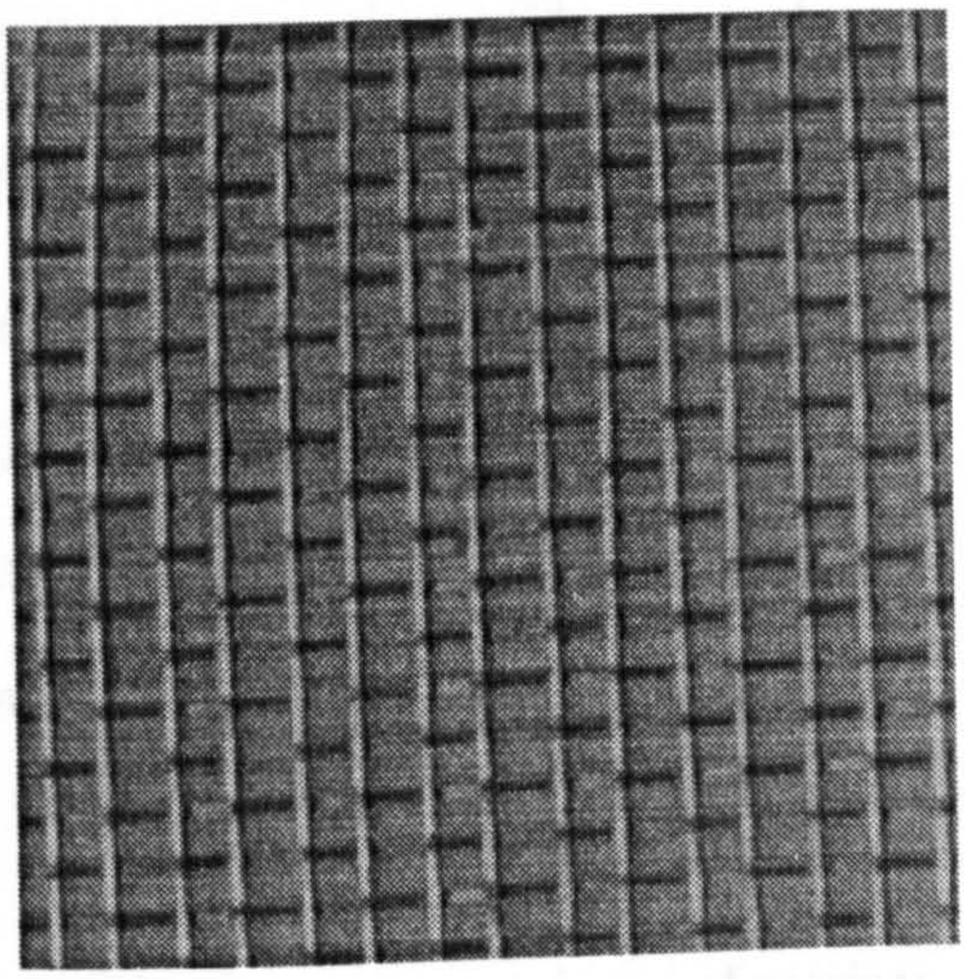
considered. The conceptual association of gray level primitives and their spatial dependence allows the evaluation of texture as having one or more of the properties such as fineness and coarseness, roughness, smoothness, contrast, directionality, and regularity. The spatial relationship among texture primitives may be random, may have a pair-wise dependence or may exhibit correlation over a neighbourhood. The dependence or correlation can be stochastic or structural. Figure 1.1(a) shows a textured image (sand) with random spatial relationships. A texture like this is often called stochastic and obeys some probabilistic laws. Figure 1.1(b) shows a textured image (wire) with its primitives organised in a regular repetitive manner. Such a texture can be called deterministic or structural. Generally, natural textures appear as a mixture of deterministic and stochastic components [81]. Figure 1.1(c) shows a natural texture (straw) with a dominating stochastic component, and Figure 1.1(d) shows a natural texture (burlap) with a dominating deterministic component.

Applications of texture analysis are enormous and amongst them the main applications can be found in the following areas [63]:

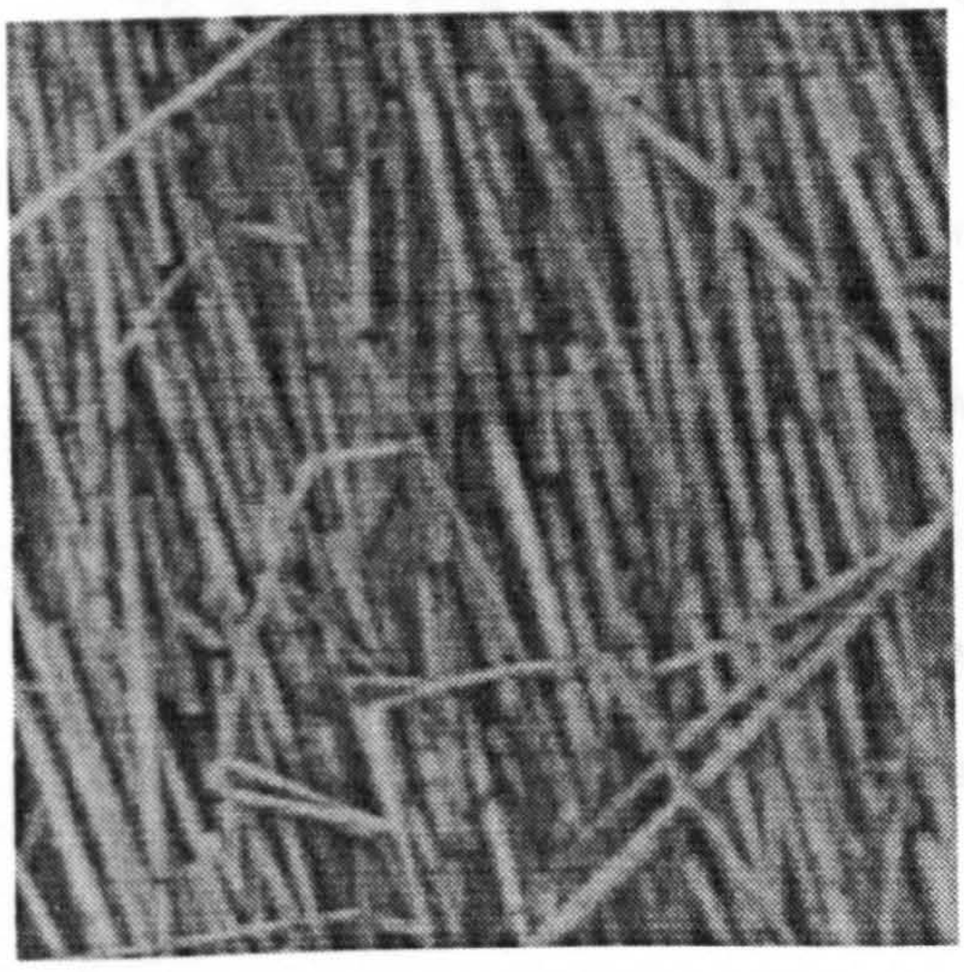
- Partitioning an image into regions of homogeneous texture
- Discriminating images based on their texture characteristics
- Extracting surface shape information from the texture disparity
- Retrieving images with similar texture from an image database
- Synthesising textures that resemble natural textures for various computer graphics applications



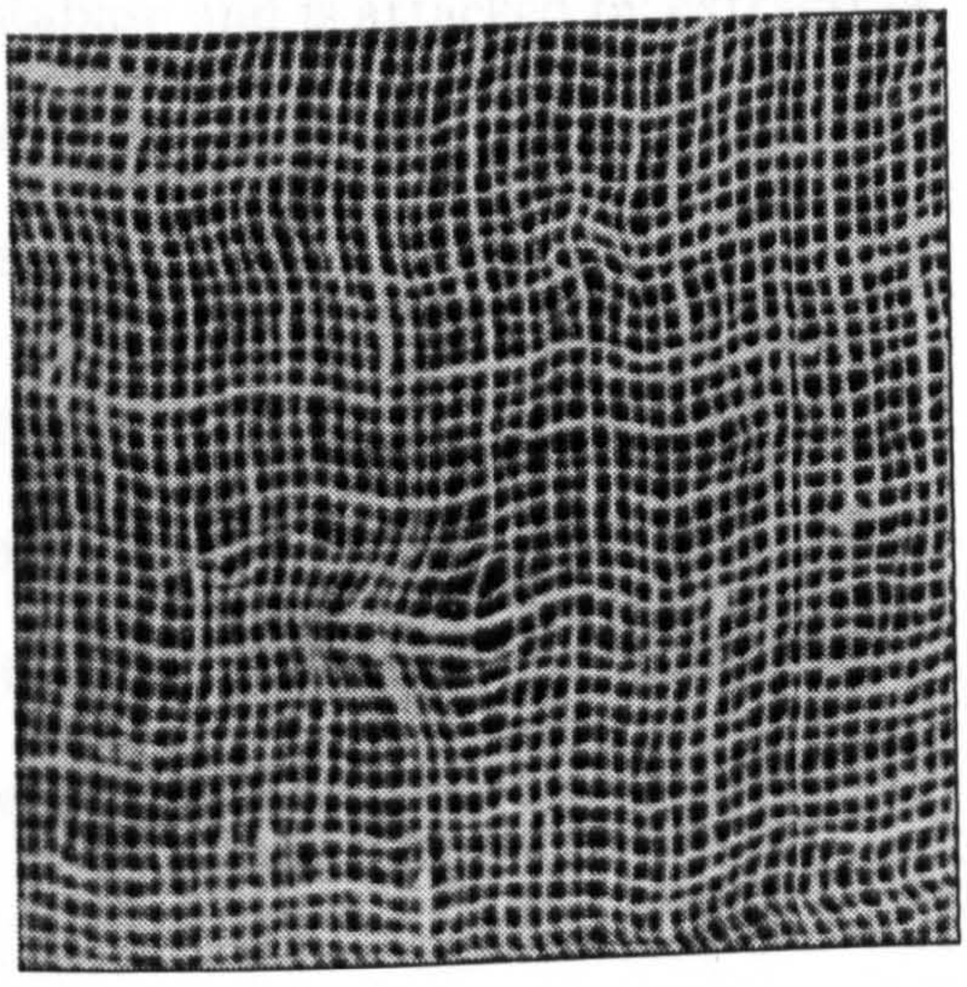
(a)



(b)



(c)



(d)

Figure 1.1: a) A textured image (sand) with random spatial relationships (b) A textured image (wire) with its primitives organised in a regular repetitive manner (c) A textured image (straw) with dominating stochastic characteristic (d) A textured image (burlap) with dominating deterministic characteristic

1.2 Issues of Texture Analysis

Texture analysis has been recognised as one of the most difficult and intriguing areas in computer vision and image processing. The main issues concerning texture analysis can be summarised as follows [34]:

1. Description and extraction of features: Given a textured region, find a description or model for it. This involves the extraction of some characteristic features of the texture region which are adequate to describe the region and to distinguish it from different textured regions.

2. Discrimination: Given a textured region, identify to which of a finite number of classes the region belongs. This is a common issue of retrieving target images from an image database and is attacked by extracting the characteristic features of the texture region and feeding this information to the classification process [81].

3. Segmentation: Given an image with several different textured regions, detect the boundaries between them. This problem is solved by, first extracting features of the different textures, followed by the application of a segmentation algorithm to detect the boundaries between different textured regions [38][80][83][86].

To deal with the first issue, a good set of texture feature descriptors should be capable of quantifying texture properties and in some applications be consistent with human visual perception, which define the visual characteristics of the given texture. Six common visual texture features suggested by Tamura [116] are coarseness, contrast, directionality, line-likeness, regularity, and roughness. To describe those features, Haralick et al. [44] suggested

some mathematical measures based on gray level spatial dependencies, such as angular second moments, contrast, correlation, variance, inverse difference moment, sum average, sum variance, sum entropy, maximal correlation coefficient. The attempt to quantify textural content and properties poses a challenge of making an appropriate description of texture and detecting features which are adequate to discriminate different types of textures present in a specific application. Inadequate sets of feature descriptors result in under-segmented results, whereas redundant sets of texture features degrade the efficiency of the algorithm without improving the segmentation quality or, even worse, downgrade the segmentation result [139].

The second issue is basically a sub-problem of the third one since an approach capable of doing segmentation is also able to discriminate textures. Application of texture analysis techniques to discrimination has found its application in the security sector, where fingerprints [84], eyes [56], and facial [41] information are analysed and the target images are retrieved from the database accordingly.

The third issue aims at the objective of partitioning a textured image into regions of homogeneous texture, each of them with its own distinctive attributes. Figure 1.2 shows an image with two different textured regions. The task of segmenting this image is to detect the boundary dividing the two textured regions based on the texture features extracted from the image.

Segmentation techniques have been practised extensively in the remote sensing [134], computer vision [103], and medical imaging [115] communities. However, the difficulties encountered in the attempt to segment images are so intractable as to amount to quasi-philosophical problems. Marr expressed his view as ([89], page 272) :

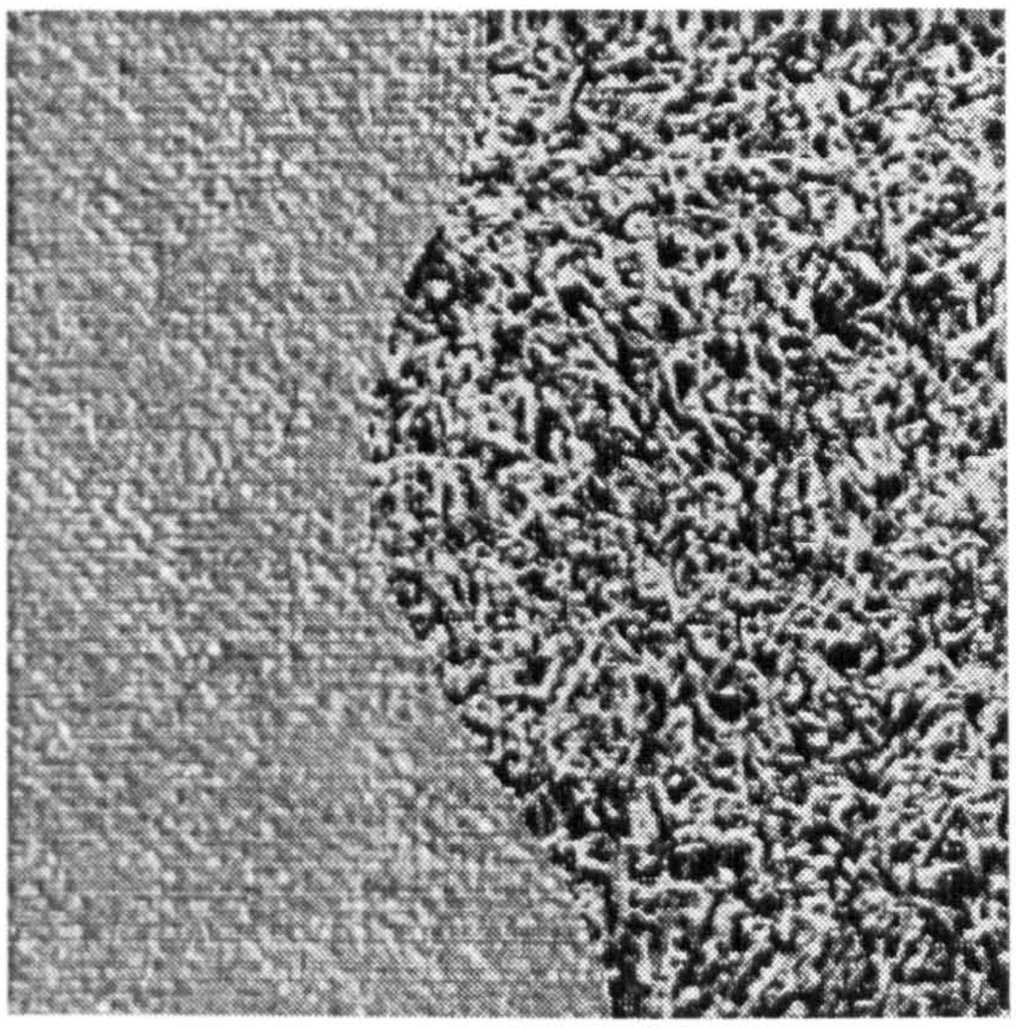


Figure 1.2: An image with two textured regions

"... , the theory and practice of segmentation remained primitive for two reasons. First, it was well nigh impossible to formulate precisely in terms of the image or even of the physical world what the exact goals of segmentation were. What, for example, is an object, and what makes it so special that it should be recoverable as a region in an image? Is a nose an object? Is a head one? Is it still one if it is attached to a body? What about a man on horseback?"

1.3 Objectives of This Thesis

Most attempts to segment or classify textures are based on either statistical or structural descriptions [45]. Statistical approaches such as co-occurrence matrices and auto-regressive models represent texture by statistics extracted from local image measurements. Generally, they are good for textures with random spatial arrangements such as the patch of sand shown in Figure

1.1(a). Structural approaches consider a textured image as composed of repeating texture elements like a piece of wire shown in Figure 1.1(b). It is clearly preferable to combine both types of feature descriptors so as to describe both statistical and structural textures and give the classification or segmentation algorithms more adequate information to carry out their tasks.

Conventional methods of segmentation generally use either region or boundary processing. In region-based segmentation algorithms [55][83][86], the manner in which initial regions are formed, the initial seeds are selected, and the criteria for splitting and merging regions are usually defined *a priori*, so that the segmentation result will rely on the choice of the initial regions and the regions' shapes will depend on the chosen growth algorithm. Also, since region-based approaches seek localisation in class space, the confidence on the boundaries' location is often weakened by the adoption of a large neighbourhood or window. Boundary based segmentation algorithms often attempt to map gray intensity space into feature space and then detect discontinuity or texture gradients in feature space by convolving a set of pre-defined masks with the feature map [68][128]. However, boundary-based segmentation algorithms, usually have difficulty in connecting boundary segments and are sensitive to intensity fluctuations within textured regions and noise, which calls for post-processing by some form of smoothing or thinning. It is clearly desirable to combine both approaches to overcome their inherent limitations.

Although some texture segmentation algorithms [8][21][25][48][55][67][86] carry out their task without supervision, most of them cannot work without intervention or supervision. A supervised segmentation algorithm requires a training phase based on a set of training textures in order to estimate the

parameters or feature vector with respect to each of the training textures. The drawback of this is that when a new texture outside the training set is given, this texture may be misclassified as one of the textures in the training set. Another drawback of supervised segmentation algorithms is that the number of the texture classes within the image of interest often has to be specified. Another major difficulty of texture segmentation resulting from the regional nature of texture is the compatibility of texture symbol (what class is the texture) with the position symbol (where is the texture boundary) [133]. If we take a big area of a textured image into consideration by using a large analysis window, we get high confidence of which class of texture this area contains. However, we lose confidence in where the texture boundary may be. On the other hand, if a smaller analysis window is utilised, the confidence in boundary position increases at the expense of compromising the certainty of texture class. This problem calls for remedy and poses a challenge to all segmentation algorithms [133][138].

Recognising the issues argued above and the need to tackle them , the objectives of this thesis are:

1. to introduce a set of texture feature descriptors which combines a structural (deterministic) component based on the affine deformation of a patch of texture [49] and a statistical component based on the local Fourier energy spectrum,

2. to propose a texture segmentation algorithm using a Markov Random Field (MRF) model, which integrates traditional region-based and boundary-based texture algorithms to overcome the limitations of both,

3. to enable the segmentation to perform its task without supervision, in

the following sense:

- the number of texture regions or classes in the input image need not to be known in advance
- the algorithm does not require *a priori* knowledge of the textures, so a learning or training phase is not needed, i.e., the texture parameters or features are estimated without supervision,

4. to demonstrate how a multiresolution or multi-scale approach can minimise the class-position uncertainty [113][133].

1.4 Outline of This Thesis

The rest of this thesis is divided into five chapters to treat the issues raised thus far. Chapter 2 gives a general overview of frequently used approaches to texture feature extraction and segmentation of textured images. Advantages and disadvantages/limitations are also compared between the different approaches.

Chapter 3 is devoted to the theoretical description of a Multiresolution Markov Random Fields (MMRF) model and stochastic relaxation which simulates the annealing process in heat treatment of materials. A general image modelling tool, Multiresolution Fourier Transform (MFT) [132], developed in the Department of Computer Science at the University of Warwick, is also briefly introduced. Some application examples of MMRF's are given and issues about the application of MMRF's are addressed. The purpose of this chapter is to provide the theoretical background for Chapter 4 and 5.

The first part of Chapter 4 introduces a novel technique for texture feature extraction, based on a two-component texture model, with a structural

component and a stochastic one [49]. The second part of Chapter 4 introduces a robust unsupervised region-based texture segmentation technique using Markov random fields (MRF) at multiple resolutions and stochastic relaxation to maximise the likelihood of texture classes. Some experiments are carried out to test the technique.

Chapter 5 presents a technique for texture boundary refinement utilising boundary information between texture regions and using the same framework as the one introduced in the second part of Chapter 4. In this chapter, the region-based approach introduced in Chapter 4 is integrated with a boundary-based approach. Some experiments are also carried out to test this technique and to show the improvement of segmentation result.

Chapter 6 concludes this thesis and suggests some future work to be done, to extend the scope of the new algorithm.

Papers of the author's published during the course of this work are listed in an appendix.

Chapter 2

Approaches to Texture Analysis

Analysing textured images is recognised as one of the most difficult tasks in image processing. The approaches to analyse them by and large remain ad hoc. Efficient segmentation methods for detecting edges or lines have failed to produce satisfactory results in analysing textured images because, in most cases, the fundamental texture elements consist of edge or line segments which are wrongly taken as boundaries, while there are not necessarily any prominent edges or lines corresponding to the real boundaries between different textured regions.

Based on the discussion in Section 1.2, it is clear that extracting features is the first step to achieve discrimination and segmentation of texture. In the rest of this chapter, a general survey of the frequently used approaches to texture feature extraction and segmentation of textured images is presented. The purpose of this survey is to review some of the representative works from different categories in texture analysis, so as to evaluate their advantages and limitations and to find a better way for texture analysis which maintains the advantages of others, but avoids their limitations, as much as is possible.

2.1 A Review of Feature Extraction Methods

Haralick [45][46], Gool et al. [40], and Reed and du Buf [105] have made excellent surveys on different approaches to texture feature extraction. These approaches can be classified into feature-based, model-based, and structural classes [42][105]. However, the distinction cannot always be clearly made and a combination of approaches from different categories is frequently adopted. The main distinction between feature-based and model-based texture analysis is that in feature-based texture analysis, texture features are evaluated without an ideal or model texture in mind, while in the case of model-based approaches, there is an underlying mathematical model which allows features to be measured by fitting the model to the texture.

2.1.1 Feature-based Approaches

In feature-based approaches, properties of textures are derived from statistical measurements, from the operation of filters, or from transformations.

A. Statistical Approaches

Statistical approaches attempt to extract statistical information such as intensity, smoothness, coarseness, and so on from the texture regions. A widely used first order statistic is mean gray level. Spann and Wilson [112] used the mean gray level of each node in a quad-tree to characterise the node.

One of the second order statistical methods uses the *autocorrelation function* to characterise the inter-relationship between pixels in textures. The autocorrelation function is capable of indicating the coarseness and regularity of a texture. If the primitives of a texture are relatively small (finer texture), the autocorrelation function will drop off rapidly with distance. If the prim-

itives of a texture are relatively large (coarser texture), the autocorrelation function will drop off slowly [45]. A high energy peak in the autocorrelation function is a good indication of regular texture. The autocorrelation methods are based on spatial averaging of second-order interactions. Kaizer [60] conducted an experiment to investigate the relationship between the autocorrelation function and textures. By assuming the autocorrelation function was circularly symmetric and computed as a function of radial distance, he found that for any smooth gray level surface, there exists a reference resolution at which no texture can be detected. As the resolution increases, finer textures are picked up. Nevertheless, autocorrelation methods are not sufficient to describe a texture since, first, many different natural textures have similar autocorrelation functions and, second, some textures of identical first-order statistics but different second-order statistics are indistinguishable [59][113]. Rosenfeld and Troy [107] also showed that autocorrelation is not a satisfactory measure of coarseness of texture. Another drawback of autocorrelation methods is that they often require large spatial windows, which results in losing the position information of texture boundaries.

The *gray level co-occurrence matrix* is another common statistical approach, first used by Julesz [57]. Haralick, Shanmugam, and Dinstein [44] specified a gray level co-occurrence in a matrix $P(i,j,d,\theta)$ as follows. Given an image $f(x,y)$ of size $L_r \times L_c$ with a set of N_g gray levels, define the matrix $P(i,j,d,\theta)$ as

$$\begin{aligned} P(i,j,d,\theta) \;=\; card\{\;& ((x_1,y_1),\,(x_2,y_2)) \in (L_r \times L_c) \times (L_r \times L_c) \mid \\ & (x_2,y_2) = (x_1,y_1) + (d\cos\theta, d\sin\theta), \\ & f(x_1,y_1) = i, f(x_2,y_2) = j,, 0 \le i,j < N_g\} \end{aligned} \quad (2.1)$$

where d denotes the distance between pixels (x_1, y_1) and (x_2, y_2) in the image, θ denotes the orientation aligning (x_1, y_1) and (x_2, y_2), and $card\{\cdot\}$ denotes the number of elements in the set. An example is given in Figure 2.1. Figure 2.1(a) is an image of size 6×6 with 5 gray levels. The corresponding gray level co-occurrence matrix with $\theta = 0$ and $d = 1$ is demonstrated in Figure 2.1(b). They suggested 14 statistical measures estimated from the gray level co-occurrence matrix to characterise textures, but Conners and Harlow observed that only five of these measures are normally useful, namely, *energy*, *entropy*, *correlation*, *local homogeneity*, and *inertia* [26]. The advantage of the co-occurrence method is that it characterises the spatial interaction between gray levels in a way that is invariant under monotonic gray level transformations. Successful applications of co-occurrence matrices have been reported in [9][16][97][140]. However, generating co-occurrence matrices can be computationally expensive, especially when the range of gray levels is large.

1	1	1	0	0	0
1	1	1	0	0	0
3	3	4	4	2	2
3	3	4	4	2	2
3	3	4	4	4	4
3	3	3	4	4	4

(a)

P(i,j,1,0)

gray level	0	1	2	3	4
0	8	2	0	0	0
1	2	8	0	0	0
2	0	0	4	0	2
3	0	0	0	10	4
4	0	0	2	4	14

(b)

Figure 2.1: Generation of gray level co-occurrence matrix. (a) Original image (b) Gray level co-occurrence matrix $P(i, j, 1, 0)$ based on (a)

Another class of statistical approaches is fractal methods. Mandelbrot [85] introduced the term *fractal* derived from the Latin word *fractus* meaning irregular segments and stated that self-similarity is an important property of surfaces. Researchers have produced a number of interesting works using fractal dimension to characterise textures, especially the roughness and self-similarity of textures [61][66][72][93][99][109][110]. The fractal dimension of textures is well suited for describing the relative behaviour of texture at multiple spatial scales. Mandelbrot [85] suggested that, given a bounded set S in Euclidean n-space, the set is self-similar if S is the union of K_r different non-overlapping duplicates of S, each of which is similar to S scaled down by a ratio $1/r$ of S. The fractal dimension D of S is then defined as

$$D = \frac{\log(K_r)}{\log(r)} \tag{2.2}$$

Now the task of feature extraction is twofold: first, the value of K_r must be found for each scale (r), secondly the fractal dimension D derived from the log-log plot of K_r versus r. The fractal dimension is then used to characterise the texture.

Sarkar and Chaudhuri [109] proposed a *box counting method* to estimate K_r in Equation (2.2) and to solve for D. Consider an image as 3-D space with (x, y) denoting the coordinates of a 2-D plane and z denoting the gray level of the pixel at position (x, y). Now if an image of size $M \times M$ is scaled down to a size $m \times m$, where m is an integer greater than 1 and the range of gray level is scaled down from G to g by the same ratio, then the ratio r in Equation (2.2) is M/m and the planar space (x, y) is partitioned into squares of size $m \times m$ pixels with (i, j) indicating coordinates. On top of each square (i, j) is a stack of boxes, each of volume $m \times m \times g$ and numbered increasing from the bottom. Let the maximum and minimum gray level of the pixels

within the (i, j)th square fall in boxes a and b respectively, then a variable $k_r(a, b)$ is assigned the value $a - b + 1$, and K_r is calculated as

$$K_r = \sum_{i,j} k_r(i, j) \tag{2.3}$$

This idea is demonstrated in Figure 2.2 where an image of size 20×20 pixels is scaled down to the size of 4×4 pixels (ratio $r = 5$) and the gray level is also scaled from 20 down to 4. Since the maximum gray level falls in box 3 while the minimum falls in box 2, $k_r(2, 2) = 3 - 2 + 1 = 2$.

Counting K_r for different scaling ratios r and using Equation (2.2), the fractal dimension D can be estimated from the least square linear fit of $\log(K_r)$ versus $\log(r)$. However, fractal dimension is only one feature of textures, using it alone is seldom sufficient to discriminate textures.

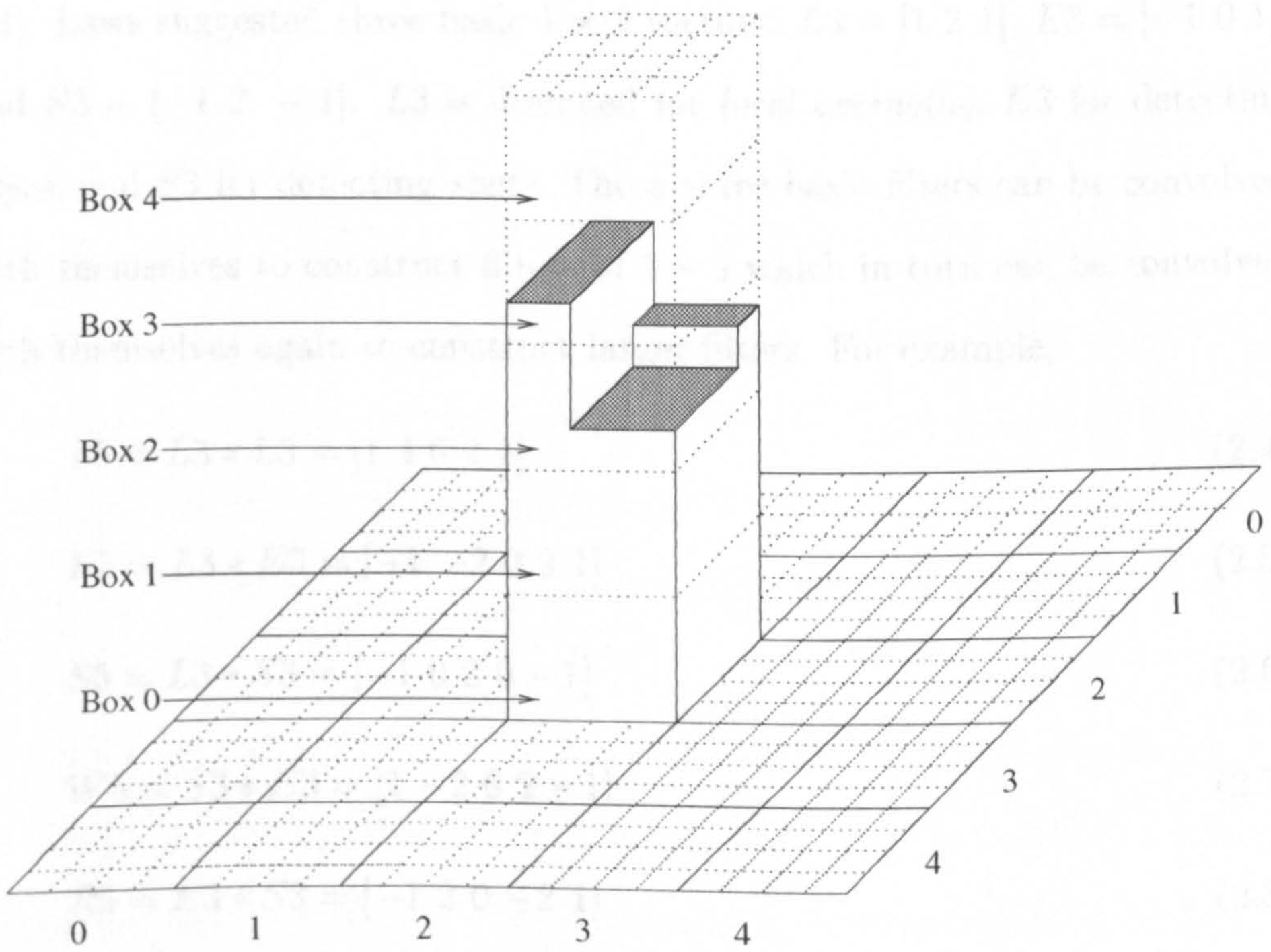


Figure 2.2: Determination of k_r.

B. Filter-based Approaches

The spirit of this method is convolving filters or operators with a textured image to derive features. One of the simplest examples was introduced by Dinstein et al. [30]. They applied a $K \times K$ filter to an image to produce an output image in which each pixel is assigned a value of the difference between the maximum and minimum gray level within the filter centred at that pixel. A high value of a pixel in the output image suggests the presence of texture and low values suggest that the pixel is contained in a homogeneous region. We can see that the success of this method relies on the value of K, the window size. Small values of K can only work satisfactorily on the textures consisting of small primitives or textures with high-frequency content.

One of the most widely adopted set of filters is the one proposed by Laws [74]. Laws suggested three basic 1×3 vectors: $L3 = [1\ 2\ 1]$, $E3 = [-1\ 0\ 1]$, and $S3 = [-1\ 2\ -1]$. $L3$ is designed for *local averaging*, $E3$ for detecting *edges*, and $S3$ for detecting *spots*. These three basic filters can be convolved with themselves to construct filters of 1×5 which in turn can be convolved with themselves again to construct larger filters. For example,

$$L5 = L3 * L3 = [1\ 4\ 6\ 4\ 1] \tag{2.4}$$

$$E5 = L3 * E3 = [-1\ -2\ 0\ 2\ 1] \tag{2.5}$$

$$S5 = L3 * S3 = [-1\ 0\ 2\ 0\ -1] \tag{2.6}$$

$$W5 = S3 * E3 = [1\ -2\ 0\ 2\ -1] \tag{2.7}$$

$$R5 = E3 * S3 = [-1\ 2\ 0\ -2\ 1] \tag{2.8}$$

where "$*$" is the operator symbol for convolution. $L5$ is constructed for local averaging, $E5$ for detecting edges, $S5$ for detecting spots, $W5$ for detecting

$L3_T \times L3$			$L3_T \times E3$			$L3_T \times S3$		
1	2	1	-1	0	1	-1	2	-1
2	4	2	-2	0	2	-2	4	-2
1	2	1	-1	0	1	-1	2	-1

$L3_T \times L3$			$L3_T \times E3$			$L3_T \times S3$		
-1	-2	-1	1	0	-1	1	-2	1
0	0	0	0	0	0	0	0	0
1	2	1	-1	0	1	-1	2	-1

$L3_T \times L3$			$L3_T \times E3$			$L3_T \times S3$		
-1	-2	-1	1	0	-1	1	-2	1
2	4	2	-2	0	2	-2	4	-2
-1	-2	-1	1	0	-1	1	-2	1

Table 2.1: The nine 3×3 filters of Laws

wave, and $R5$ for detecting *ripple*. By utilising vector multiplication, $n \times n$ centre-weighted filters can also be obtained. Table 2.1 illustrates the nine filters constructed by the multiplications of the three basic 1×3 filters of Laws. It is interesting to note that Sobel operators are among this set of filters.

Laws's approach to feature extraction is a two-stage process which involves first measuring of microfeatures and afterwards calculation of macrofeatures as texture features. The microfeatures are measured by convolving a $n \times n$ filter with the original image, with the resulting image indicating local edginess, spots, etc.. referred to as the feature image. In the second stage,

the local statistics of these properties in the feature image are computed and then smoothed with a window significantly larger than the filters used in the first stage (e.g. a 15 × 15 smoothing window is used in Laws's work after applying 3 × 3 filters). The most useful local statistics are the sums of the squared or absolute values of the microfeature within the smoothing window which Laws termed *texture energy measures.*

Laws's approach has influenced many subsequent works by different researchers [21][48][96][100][121][126]. Patel et al. [96] adopted Laws's approach to detect foreign objects (defects) in food. They used 3 × 3 Laws filters to obtain microfeatures, the macrofeatures were then calculated using the sum of absolute value of microfeatures within a 5 × 5 window to create texture energy images.

Although Laws's approach is computationally efficient, since the size of the filters is relatively small, only the high-frequency content of textures can be characterised and the approach is essentially ad hoc. Pietikäinen et al. [100] argued that the power of Laws filters depend on the general form (spot-like, edge-like, etc.) rather than on the pre-set numerical values of the filters. Ade [2] thus proposed a method marrying Laws's and co-occurrence matrix approaches to automatically generate convolution filters, depending on the texture. These automatically generated filters are called eigenfilters, which were derived from eigenvectors of the principle component decomposition of the covariance matrix of the local neighbourhood. The advantage of this approach is that it reduces the number of filters needed for discrimination.

C. Transform-based Approaches

In transform methods, images are usually divided into overlapping or non-overlapping blocks, each of which is then transformed into a new coordinate system of a different domain. Frequency is so closely related to texture that researchers often use transform methods to transform images into the spatial frequency domain for extracting features.

Perceptual experiments of Julesz [58] and others [5][14] have suggested that 'effortless' discrimination of texture is generally only achievable when their spectral attributes differ significantly. Chen [19] has explained that the claimed improvement in performance of co-occurrence based methods over simple spectral methods reported in the work of Weszka et al. [130] is due to crudeness of their spectral estimation techniques. Chen [19] reported that using a more accurate method, based on maximum entropy, results as good as the methods based on co-occurrence matrix can be obtained. Xu and Fu [136] used the Walsh transform as a measure of texture coarseness for segmentation. D'Astous and Jernigan [28] derived a set of texture features such as peak strength, two-dimensional curvature, area, squared distance from the origin, and angular location from the peaks of power spectrum, and another set of texture features such as spread, circularity, and elongation from the shapes of power spectrum.

A method using multichannel Gabor filters to transform the image into the frequency domain for extracting texture features by analysing the spectrum was first proposed by Knutsson and Granlund [70] and has been widely adopted since [8][10][32][33][55][101][104][114][118][127]. The goal is to transform texture differences into detectable filter-output discontinuities at texture boundaries. This multi-channel filtering approach decomposes an image

with a bank of filters tuned to a specific combination of frequency and orientation, into several filtered images each with limited spectral information. The decomposition allows the exploitation of differences in dominant sizes and orientations of different textures. Another advantage of this approach is that it allows simple statistics of gray levels in the filtered images to be used as texture features. Knutsson and Granlund [70] used a set of four log-normal quadrature filters to estimate the local texture orientation. With this set of filters, structural properties of the texture are mapped into a slowly varying vector field representing local orientation. They also proposed three sets of quadrature filter pairs to estimate the local texture spatial frequency [42]. Each set of filters has a different centre frequency and each filter in the same set has a directional vector orthogonal to the one of the other filter. They use the estimated local orientation and frequency for texture segmentation. Wilson and Spann extended this approach using a complete transform based on Prolate Spheroidal Sequences [133].

Jain and Farrokhnia [55] used 28 real-valued, even-symmetric Gabor filters to characterise the multiple channels. The impulse response of an even-symmetric Gabor filter is given by [35]

$$h(x,y) = exp\left\{-\frac{1}{2}\left[\frac{x^2}{\sigma_x^2}+\frac{y^2}{\sigma_y^2}\right]\right\} cos(2\pi u_0 x) \tag{2.9}$$

where u_0 is the frequency of a sinusoidal wave function along the x-axis, and σ_x and σ_y are constants of the Gaussian envelope along the x and y axes, respectively. Filters with arbitrary orientations are obtained by rotating the x-y coordinate system. They first convolve the original image with each of the bank of Gabor filters to generate filtered images; each of these was then subjected to a nonlinear transformation to capture the attributes of blobs detected in the Gabor filtered images. Subsequently, the average absolute

deviation from the mean in small overlapping windows was used as texture features for each pixel in the nonlinearly transformed images. One of the limitations of this approach is the lack of a criterion for choosing the value of the parameter in the non-linear transformation. Also the approach assumed that different channels are independent from each other, which is contradict to the psychophysical and physiological evidence that there is interaction between different spatial frequency channels [123].

Generally, one disadvantage of Gabor filters is their non-orthogonality, which leads to redundant features at different channels or scales. It was shown that down-sampling the Gabor transform always results in severe degradation of segmentation quality [102].

Among the transform-based approaches, one of the most popular ones in recent years uses a filter bank as the wavelet transform to decompose a textured image into sub-band images from which the texture features are then extracted [4][20][73][75][83][87][122][137]. Hsu and Wilson [50] used the spectral energy density estimated in each block via the Multiresolution Fourier Transform (MFT) [132], a generalised wavelet transform, to characterise texture in the hope that some textures have a distinctive spectral energy density which allows texture differentiation [49]. Chen and Kundu [20] suggested a method to identify textures by combining quadrature mirror filter (QMF) banks and hidden Markov models (HMM). First, a QMF bank is used as the wavelet transform to decompose the textured image into sub-band images. Features such as normalised energy and normalised entropy are then extracted from the sub-band images by deriving the statistics based on the first order probability distribution of gray levels. Secondly, the sequence of sub-bands is modelled as a hidden Markov model (HMM), and one HMM is

designed for each class of textures to exploit the dependency among these sub-bands. During the classification process, the unknown texture is matched against all the models. They reported up to 93.33% classification accuracy. However, the algorithm requires a training process and the specification of the number of states in the HMM which limits the generality of this method.

Lu et al. [83] utilised a pyramid-structured wavelet transform to decompose an image into four sub-images of different frequency bands (low-low, low-high, high-low, and high-high). The decomposition is then repeated on the sub-image of the low-low band to obtain the four sub-images at the next scale. Afterwards, an 'energy measure' of the wavelet coefficients of the three sub-images of higher frequency bands at each scale is used as a texture feature. Denoting $W_{k,l}(i,j)$ the (i,j)th wavelet coefficient in the $n \times n$ window centred at pixel (k,l), the 'energy measure' of pixel (k,l) is defined as follow.

$$E_{k,l} = \frac{1}{n^2} \sum_{i,j=1}^{n} |W_{k,l}(i,j)| \tag{2.10}$$

This is similar to the approach used by Wilson and Spann [133], but it lacks the resolution in frequency to discriminate many textures.

2.1.2 Model-based Approaches

The aim of mathematical modelling in texture analysis is to capture the intrinsic features of the texture in a set of parameters, so as to understand the properties generating the texture. Any analytical expression explaining the properties of texture primitives and the dependency of primitives on their neighbourhood can be called a *model.* Having an underlying model, the analysis of a textured image can be more efficiently done by fitting the model to the image. Moreover, synthetic textures can be generated with an underlying texture model and compared visually with those of the data.

A. Autoregressive Models

Autoregressive models are one of the major stochastic models utilised for texture analysis [18][62][64][88][108][111]. One of the commonly used models is shown in Equation (2.11) [64].

$$l(i,j) = \sum_{(k,l)\in N_1} \phi_{k,l} \cdot l(i+k, j+l) + e(i,j) \tag{2.11}$$

where $e(i,j) = \sigma_a \cdot a(i,j)$ is a zero-mean correlated process and $l(i,j)$ is the gray level of the pixel at position (i,j). By identifying appropriate parameters of the AR models, the estimated parameters may be used to describe textures. Typically, the parameters $\phi_{k,l}$ are estimated using a likelihood-based technique.

Although this technique provides a good scheme for classification, the complexity and the computational cost of estimating the potentially large number of parameters is a serious concern. Choice of the neighbourhood N_1 is also a critical and difficult problem. Moreover, it is nontrivial to apply such modelling to segmentation, since reliable parameter estimation conflicts directly with boundary localisation.

B. Markov Random Field Models

One of the most popular models utilised for texture analysis in recent years, MRF theory allows textures to be modelled in a more general way than simple linear models. A particular MRF model favours its own class of textures by associating them with larger probabilities than other texture classes. MRF theory is usually employed in conjunction with statistical decision and estimation methods, so as to formulate objective functions in terms of established optimisation principles. Maximum *a posteriori* (MAP) probability

is one of the most popular statistical criteria for optimisation and in fact, has been the most popular choice in MRF texture modelling [37][78]. MRF's and the MAP criterion together give rise to the MAP-MRF framework advocated by Geman and Geman[37] and allow the systematic development of algorithms for a variety of vision applications.

Cross and Jain [27] defined a Markov random field using a binomial joint probability density function on a neighbourhood of gray level pixels and fitted the model to a prototype texture measuring the associated parameters. The parameters were taken as features of the prototype texture and used to generate a synthetic texture which resembles the prototype. They used a hypothesis test to evaluate the goodness-of-fit under the model and found that their approach worked well for microtextures, but not for macrotextures.

For the purpose of texture segmentation, Krishnamahari and Chellappa presented multiresolution models for Gauss-Markov random fields to represent textures at different resolutions [71]. On a 2-D lattice Ω of size $M \times N$, each class of texture is modelled as a Markov random field X. The Markovianity of the joint probability density function of X is given as :

$$\begin{aligned} p(x_s|x_t, \forall t \neq s, t \in \Omega) &= p(x_s|x_{s+r}, r \in \eta) \\ &= \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{ -\frac{\left[x_s - \sum_{r\in\eta} \theta_r x_{s+r}\right]^2}{2\sigma^2} \right\} \end{aligned} \tag{2.12}$$

where η is the neighbourhood of site s, σ and θ_r are the model parameters which describe the texture features. Sub-sampling Ω k times results in an image pyramid of $k+1$ levels with different resolution. The MRF at each level of the image pyramid has it own parameters set $\sigma(k), \theta_r(k)$. However, Markovianity is lost at coarser resolutions under such a sub-sampling operation. They presented two different parameter estimators, *Kullback-Leibler*

distance minimisation and *local conditional distribution invariance approximation* to obtain a Markov approximation.

In [86], Manjunath and Chellappa divided an image into a number of non-overlapping regions, each modelled by a Markov random field. Associated parameters of each MRF are estimated from the corresponding region using a least squares estimate [65] and assigned as a feature vector to represent that region. A simple clustering method was then used to perform coarse segmentation based on a distance measure between pairs of feature vectors. After the coarse segmentation, the parameters of the MRF's were recomputed and then used in a pixel based fine segmentation algorithm involving an approximation to the MAP estimate of the texture labels. The disadvantage of this technique is the simplicity-stability tradeoff. To reduce the computational cost, they used a least squares estimate [65] instead of an optimisation algorithm to obtain the estimate of the parameters, with the result that the requirement that the covariance matrix for the joint probability density of their texture model being positive definite, is not guaranteed.

Generally, MRF theory allows textures to be modelled in a more general way, however, it often requires large computational resources, especially when stochastic relaxation is adopted, and convergence rate can be a problem.

2.1.3 Structural Approaches

Structural approaches are utilised to analyse deterministic textures consisting of similar primitives spatially arranged according to some set of well-defined placement rules. However, in reality, natural textures seldom consist of identical primitives grouped by rigid placement rules, so that structural approaches are not frequently applied. Some structural methods of feature

extraction start from extracting texture primitives and then the spatial placement rule is analysed [91][119][120]. Methods working in this style are sometimes called *bottom-up*. In [91], Matsuyama et al. proposed an approach combining bottom-up and top-down phases, which first extracts texture elements by executing a region-growing procedure based on gray levels. A clustering procedure then follows to extract 'regularity vectors' from a set of relative positional vectors which appear often between the texture elements detected. False elements detected in the region growing can also be rejected by the clustering procedure since false elements usually appear randomly in the image and as a result the number of regularity vectors similar to those from/to the false elements is much smaller than the number of regularity vectors between real texture elements. Up to this stage, the algorithm is basically performed bottom-up and some texture elements may be missed out because of the presence of noise. Thus a top-down phase is performed to locate possible missing elements by applying a template matching method based on regularity vectors. Once the false elements are rejected and the missing ones detected, the regularity vectors are analysed to remove the redundant ones, which are the linear combination of other simple ones. The simple regularity vectors are then used to describe the spatial placement of texture elements.

However, methods of this type are often sensitive to various image degradations such as shading, blurring, random noise, geometric warping, etc., so that their application is limited to rather simple and synthetic textures. In order to circumvent this limitation, Matsuyama [90] proposed a *top-down* method by which two spatial vectors representing the periodicity (placement rules) and the phase information are first estimated from the energy distribu-

tion of the power spectrum of the texture. Afterwards, with a region-growing method based on the spatial vector pairs and the phase information, texture primitives are extracted. The implicit assumption of this method that the texture consists of absolutely periodic primitives inevitably imposed a limitation on its application to most natural images, where texture primitives are usually irregularly shaped, or even randomly grouped without a clear shape. Another disadvantage of this method is that when the texture primitive itself has a significant directionality of its own, which is different from that of the displacement direction, there will be significant energy along this direction as well as along the displacement direction. This may result in taking the wrong spatial frequency as the representative frequency.

Mathematical morphology models extracting texture features from morphological residues of opening and closing are frequently used by researchers [23][31][79][124][125]. Dougherty at el. proposed morphological granulometries as filter sequences to describe textures [31]. Based on the property of *opening* (we use $\circ$ to represent the operator of opening) that if $F \circ E = F$, then for any set S, $S \circ F < S \circ E$ and F is called E-open, an image sequence $S \circ E_k, k = 1, 2, 3, \cdots$, each called *granulometry*, is created by subjecting image S to the operation of *opening* by a sequence of structuring elements E_k for which E_{k+1} is E_k-open. Counting the number of pixels remaining in each granulometry, a decreasing function $\Psi(k)$ called *granulometric size distribution function* results. The derivative of the normalised $\Psi(k)$, denoted by $d\Psi_n(k)$, is now frequently referred to as *pattern spectrum* and the moments of $d\Psi_n(k)$ are employed to extract texture features. To utilise this theory for texture segmentation, Dougherty at el. thresholded each image into a binary one and placed a window $W(i, j)$ at each pixel (i, j) of the image and

calculated the *pattern spectrum*, $d\Psi(i,j)$, by operating *opening* within the window. They employed a sequence of circular structuring elements of varying radii and four sequences of structuring elements of one-pixel-wide and varying length, namely vertical, horizontal, +45° diagonal, and −45° diagonal. After the pattern spectra corresponding to each of the five structuring element sequences are calculated for each pixel, their first three moments are computed to form a texture feature vector. Using these feature vectors, each pixel is classified using a Gaussian maximum likelihood scheme. The weakness of this scheme is the essentially ad hoc nature of the feature extraction.

Huet and Mattioli [53][54] proposed a mathematical morphological method to extract a minimum family of structuring elements, called a *minimal generating basis*, to represent texture primitive patterns (texels). They convolve a window (from 2×2 to 10×10 pixels in most cases) with a textured image (taken as a sample for learning) and extract the minimal generating basis from the set of all neighbourhoods, each one corresponding to a window and belonging to the invariance domain of the morphological transformation. Since a structuring element of the minimal generating basis corresponds to a primitive pattern (texel), the more complex the texture, the greater the number of structuring elements in the minimal generating basis. They obtained some success in application of this method to defect detection and recognition of textures. They observed that the precision of the texture characterisation is proportional to the structuring element size. It may be expected that the computational cost will increase significantly when textures with high complexity are involved and the robustness of the method is limited by the sample space of textures used in the training process and the presence of noise.

Chen et al. [24] proposed a set of sixteen texture features based on the statistics of geometrical properties measured from a textured image. They first generate a set of binary images by thresholding the original image with a set of different threshold values. In the second step, for each binary image, pixels with their gray level above the corresponding threshold are grouped into connected regions. The pixels with gray level below corresponding threshold values are also grouped into connected regions in the same way. The *irregularity* of the xth connected region of the binary image with threshold value α defined as following is then calculated. Sixteen measures: *maximum value, average value, mean value*, and *standard deviation* of the number of connected regions within the binary images with gray level above and below the thresholds, the *irregularity* of connected regions and the *average irregularity* with the binary images are used as features for classifying textures. They compared this method favourably with Spatial Grey Level Dependence Matrix (SGLDM), Liu's features [82], and Statistical Feature Matrix (SFM) [135] on the complete 112 textures set from Brodatz [12]. The correct classification rate for this method is 85.6%, 64.6% for SGLDM, 32.7% for Liu's features, and 72.8% for SFM. However, for some textures with patterns much larger than the size of the window used, misclassification is inevitable. The method seems, like many others based on morphology, to have an ad hoc or heuristic basis. Also, as the authors reported, this method requires even more computational time than the time-consuming co-occurrence matrix approach.

Fourier methods have also been used in characterising the structure of textures. After examining the Fourier spectrum of all the images in the Brodatz database, Liu and Picard [81] concluded that :

- Perceptually structured textures usually have dominant harmonic components which appear as structured spectral peaks. Conversely , when the harmonic components are strong, they usually dominate the perceptual pattern discrimination.

- Although certain local inhomogeneities (such as texture on an uneven surface or viewpoint distortion) spread out or change the frequencies of the spectral peaks slightly, the intrinsic structure of these peaks remains.

- Strong evanescent components correspond to prominent directionality in patterns; local inhomogeneities have only a minor effect on these components.

Based on the above conclusions, they put forward a Wold-based texture model for describing textural features [81]. Before feature extraction is attempted, they first carried out a *harmonicity test.* The auto-covariance function of an image is calculated as the inverse discrete Fourier transformation (DFT) of the image power spectrum. For highly structured textures the auto-covariance energy is concentrated periodically throughout the 2-D displacement plane. A region growing outwards continuously from the zero displacement until the value of the auto-covariance function drops to a specific level (10 % of the auto-covariance function range in [81]) is identified. This region is called the *small displacement region.* For each image in the Brodatz database, the ratio between the energy in the small displacement region and the total energy of the image, r_e is calculated. The histogram of the ratios is bimodal: the more structured the texture, the higher the ratio. Denoting the two classes as c_h (harmonic) and c_r (random), the posterior

probability of c_h is calculated as

$$p(c_h|r_e) = \frac{p(r_e|c_h)p(c_h)}{p(r_e|c_h)p(c_h) + p(r_e|c_r)p(c_r)} \tag{2.13}$$

This probability was used as the confidence measure for characterising the image as highly structured.

To extract the texture features, the mean of the image is first adjusted to zero and then the image is Gaussian tapered, followed by the computation of DFT. The local maxima of the DFT magnitudes are detected by searching a 5×5 neighbourhood and examined for the harmonic peaks. In their work they only used the ten largest harmonic peaks for the Wold feature set. In image retrieval, the matching of the texture harmonic structures is done by comparing the Wold feature sets. These feature sets inherit the property of *spatial shift-invariance* from the Fourier spectral magnitude, which is important for matching or comparing images in the image retrieving applications. However, the use of spectral peaks may be adequate for some texture, but it is not a general approach. Moreover, warping of the textures caused by imaging geometry will affect peak locations.

Hsu and Wilson [50] proposed a method of texture synthesis by identifying the affine transformation which gives the best match between pairs of texture patches of a given size. The idea is to extract a pair of representative centroid vectors of the spectrum for each texture patch and to find the 2×2 affine transformation matrix which transforms the representative centroid vectors of a prototype patch into the centroid vectors representing a test patch. The affine transformation matrix is then used to transform all the coordinates of the Fourier coefficients of the prototype into a new set of coordinates with the same Fourier coefficients. An inverse Fourier transform is then performed on the new spectrum to synthesise the test patch. To extract

the representative centroid vector pair, they first split the spectrum into two angular segments. Within each segment, the centroid vector and the variance with respect to the centroid vector are calculated. A set of centroid vector pairs, each corresponding to a different angle, separating the spectrum into two segments, can be obtained. The vector pair corresponding to the smallest variance is chosen as the representative vector pair of the texture patch. Applications of this affine transformation method are reported in [49][51][52]. It also forms the basis of the work reported in Chapter 4.

In summary, feature extraction approaches should be capable of detecting both structural and statistical features isotropically. Based on the survey carried out above it is clearly preferable to combine both statistical and structural approaches so as to achieve this objective. This is one of the goals we are to pursue in this work.

2.2 A Review of Segmentation Methods

The goal of image segmentation is to divide an image into connected regions each of which has homogeneous characteristics such as gray level or texture, preferably a perceptual property, so that the segmenting of the image is consistent with human perception. Regions should have uniform content without small holes, boundaries between regions should be spatially simple, not ragged. This task is complex and mathematically close to intractable [45][133].

Approaches to texture segmentation are roughly divided into region-based, boundary-based, and hybrid categories. Region and boundary-based approaches both seek texture features within neighbouring blocks of the target image. The main difference is that region-based methods are concerned

with feature homogeneity, while boundary-based are concerned with the feature inhomogeneity. Hybrids of the two approaches are often attempted in the scope of getting the advantages of one approach to complement the other.

2.2.1 Region-based Approaches

Among earlier region-based approaches are split-and-merge [22][47], linkage region growing [11] and clustering [94]. They all attempted to group pixels or image blocks of similar characteristics together. Spann and Wilson [133] proposed a quad-tree approach which consists of three components: quad-tree smoothing , classification and boundary estimation. The smoothing is performed by letting $d(i,j)$ be the $(N \times N)$, $N = 2^m$ field representing the data and the quad-tree is denoted as

$$\begin{aligned} q(i,j,k) &= \frac{1}{4}(q(2i,2j,k-1)+q(2i+1,2j,k-1) \\ &\quad +q(2i,2j+1,k-1) \\ &\quad +q(2i+1,2j+1,k-1)), \text{ for } k>0 \end{aligned} \tag{2.14}$$

and

$$q(i,j,0) = d(i,j) \tag{2.15}$$

The maximum smoothing gain is obtained by truncating the quad-tree at a level $m' < m$. The classification task is performed by a local centroid clustering which moves probability masses to their centre of gravity within a window of width $(2m+1)$ covering the histogram of the image to be classified. The convergence properties of the clustering algorithm depend on the window size and the 'peakiness' of the histogram. To alleviate the class-position uncertainty once the classification algorithm has converged, nodes with their class label different from any of their 8-neighbours are marked

as members of the boundary region. Nodes not in the boundary region are given the same class label as their fathers; nodes in the boundary region are classified after smoothing, in such a way that their width is reduced by a factor of two each step down the quad-tree. Thus the result of this process is an estimate of the boundary between pixels at the lowest level of the quad tree. They reported that the approach is flexible enough to deal with many cases of practical interest, including those where the signal-to-noise ratio is well below 1. However, the performance of this technique can only be expected to be good when the assumptions underlying the algorithm, such as those relating to the spatial coherence of the regions to be segmented, are justified.

In [86], Manjunath and Chellappa first divide an image into a matrix of equal sized non-overlapping windows with a texture feature vector attached to each window. Each feature vector is a set of parameters of a Gaussian Markov random field which are used to model the texture. A simple clustering method is applied to group the windows into regions in which each pair of neighbouring windows has a normalised Euclidean distance smaller than a threshold. MRF parameters are re-estimated from the coarsely segmented regions and used to refine the segmentation in a relaxation algorithm called maximum a posteriori marginal (MPM) which minimises the expected number of misclassified pixels. The limitation of this approach is that the approximate number of texture classes has to be specified and a simple heuristic which is a function of the approximate number of texture classes is adopted as the threshold for merging regions.

Jain and Farrokhnia [55] filtered an image with a set of Gabor filters with different parameters to create a sequence of filtered images.. Texture features

are then extracted by subjecting each filtered image to a non-linear transformation and computing an energy measure in a window around each pixel. They then perform a two-phase iterative clustering algorithm to achieve segmentation. Phase 1 is a K-means pass which creates a sequence of k clusters. Phase 2 then creates another sequence of clusters by merging two existing clusters at one time to seek for better clustering. After each iteration of the two phases, the square error of the new clustering is compared to the square error of the clustering of the previous iteration. This process is iterated until no further reduction of square error is possible. The limitation of this approach is the high computational cost for estimating the number of texture classes and the unsatisfactory performance of this estimation, especially when the true number of classes is large.

In [83], Lu at el. first performed a wavelet decomposition on the original image to create a low-frequency and three high-frequency channels. For each of the three high-frequency channels, energy measures of the wavelet coefficients are extracted to form a feature image. The same process is repeated on the low frequency channel so that a multi-scale hierarchy is formed. For each scale, once the three feature images are created, a multi-thresholding method is applied to partition each into regions. High resolution segmented images at each scale are achieved by an intra-scale procedure which combines the three coarsely segmented images. Afterwards, the finely segmented image of the next coarser scale is expanded so that it can be combined with the finely segmented image of current scale by a inter-scale fusion process. This inter-scale fusion process is repeated from the fine scale to the coarse scale, until a stop condition is met. The purpose of the feature fusion process is meant to determine the number of texture classes. They reported a

segmentation result of an image consists of four equal-sized square textured regions at an error rate of 4.49%. The misclassification occurred mainly along the boundaries, particularly around the corner where the four regions meet. However, the thresholds deciding the minimal distance which two regions should at least be distant and the minimal size of region are chosen ad hoc and texture dependent. Inappropriately chosen thresholds can give rise to over-segmentation or under-segmentation.

2.2.2 Boundary-based Approaches

One of the main limitations of region-based approaches is their reliance on constant properties across each region. Because of surface geometry and other sources of variation, this condition is often not met in practice. To overcome this problem, it may be advantageous to look for significant local changes in properties. This is the basis of boundary-based segmentation. Traditional edge detection algorithms are designed to detect boundaries between regions with homogeneous gray levels. These approaches are bound to fail when applied directly to detect boundaries between textured regions, due to the gray level fluctuations within textured regions. To make these edge detection algorithms work, textured images have to be transformed from gray level space to feature space.

One of the earlier approaches was proposed by Knutsson and Granlund, which detected the boundary between oriented textures using quadrature filters [70]. Wermser [128] proposed a set of four masks as shown in Figure 2.3 for convolving with a texture feature image estimated from the original image. By comparing the two opposing subregions, numbered 1 and 2, of interest of each mask, a disparity measure is obtained. Masks M_1 and M_2

are used to detect the disparities along the vertical and horizontal directions respectively, while M_3 and M_4 are used along 45° and 135° respectively. The maximum disparity measure among the four masks is attached to the corresponding pixel to form a texture gradient image and segmentation is achieved by thresholding the texture gradient image and performing line thinning to the detected boundary. Although this approach demonstrated the interesting

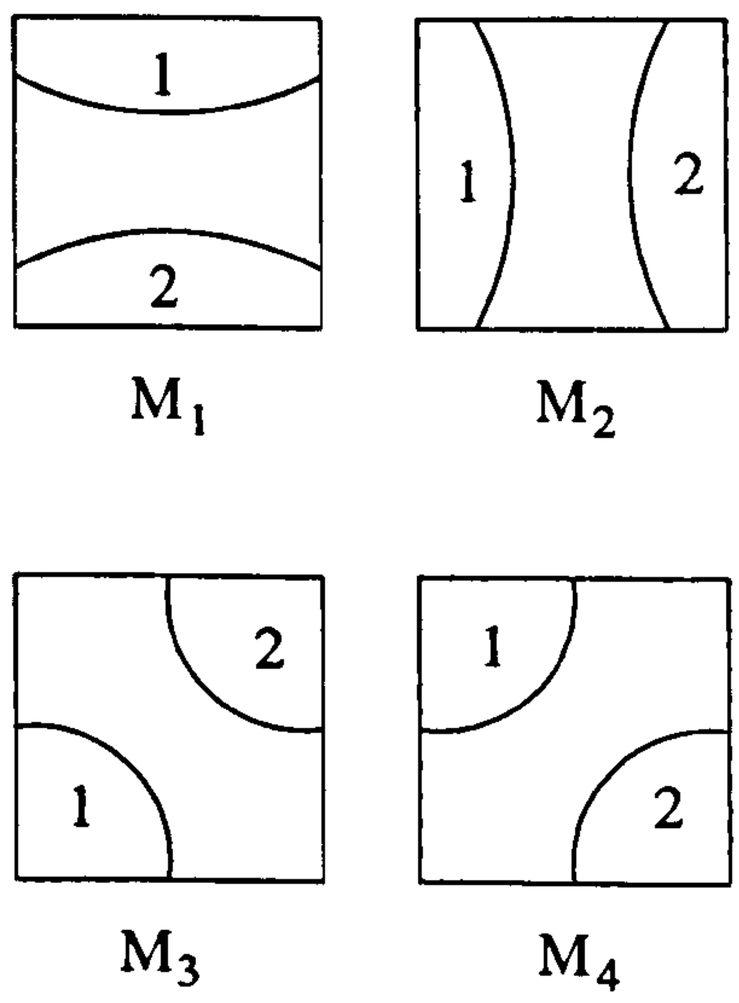


Figure 2.3: Texture masks proposed by Diederich Wermser for detecting texture gradients.

idea of texture gradient estimation, the segmentation method, relying on the simple thresholding, is not sophisticated enough to yield satisfactory results in terms of boundary continuity and accuracy.

Khotanzad and Chen [68] suggested a boundary-based approach by applying traditional edge detection operators such as Sobel to texture feature images. They first sample the original image by sliding a window in D-pixel wide steps in both horizontal and vertical directions. Six parameters of a simultaneous autoregressive (SAR) model are then extracted from each window to describe texture features contained in that window. The six features

are then normalised so that they can be integrated into a 'textural change' image. The textural change image is then transformed into a gray level image and Sobel operators are applied to detect edges. A thresholding operation is conducted to create an edge/non-edge image. Due to micro-edges within the textures, a clean-up step is performed to remove isolated edges from the edge/non-edge image. This edge/non-edge image is then mapped onto the original image by expanding each pixel of the edge/non-edge image D times in both horizontal and vertical directions. This expansion is necessary because the window slides over the image in D-pixel wide steps in both horizontal and vertical directions. The boundary contained in the expanded edge/non-edge image is D pixels wide and thus calls for thinning. They use the approach suggested in [106] to achieve the thinning. The performance of the suggested algorithm is satisfactory on detecting boundaries with no sharp corners. However, since this is a window-based technique, it is not possible for the algorithm to locate edges with pixel level accuracy. It is because of this reason that the segmentation around sharp corners is not accurate enough. Another disadvantage of this method is that continuity is not guaranteed.

Tan [117] put forward a texture edge detection algorithm which performs six successive steps: spectral peak detection, 'cortical channel' filtering, gradient estimation, channel combination, edge thresholding, and post-processing. The purpose of spectral peak detection is to determine the number of cortical channels and channel parameters. The Converging Squares Algorithm (CSA) suggested by O'Gorman and Sanderson [92] is employed to iteratively detect the spectral peaks. The cortical channel filtering is carried out on the spatial frequency domain via FFT and inverse FFT. The output

image of each channel is supposed to reveal strong texture energy concentrated in the vicinity of the central frequency and along the orientation to which the channel is tuned. Each channel output image is then convolved with Sobel operators to create a gradient image. A combined gradient image is then created by summing up all the individually weighted gradient images. This combined gradient image is subjected to a thresholding step to produce an edge image. Finally, post-processing is performed to thin the texture boundary and remove short edges. The performance of the algorithm is satisfactory except at the sharp corners where rapid textural transitions occur. Tan also reported that the algorithm can tolerate up to 20 % channel parameter perturbation without much degradation in segmentation performance. However, using only gradient magnitude, this algorithm is not capable of differentiating textures which only differ in phase. Also, since the algorithm assumes that textures have distinct spectral peaks which correspond to some global regularities, textures with stochastic or irregular characteristics cannot be discriminated.

2.2.3 Hybrid Approaches

In region-based segmentation algorithms, the manner in which initial regions are formed, the initial seeds are selected, and the criteria for splitting and merging regions are defined a priori, thus the segmentation results rely on the choices of the initial regions and the region shapes depend on the chosen growth algorithms. Also since region-based approaches seek localisation in class space, the confidence upon where the boundaries are is weakened by the adoption of large neighbourhood and window.

Boundary-based segmentation algorithms, on the other hand, usually

have difficulty in connecting boundary segments and are sensitive to noise and intensity fluctuation within textured regions, which calls for post-processing.

Integration of both region and boundary-based approaches attempts segmentation from localisation in both class space using region information and position space using boundary information, in the hope that the aforementioned shortcomings can be alleviated to some degree at a slight additional computational cost. Although this combination is clearly desirable, it remains less explored because how it can be achieved is not obvious. Also, because of the inherent difficulties in analysing textures, most of the reported approaches which incorporate region and boundary processes were only performed on gray level images. In the rest of this section, four interesting approaches which incorporate region and boundary processes are introduced [3][7][98][138]. Although only [138] is performed on textured images, the others also provide interesting aspects which are possible to be adapted to segment textured images.

Zhu and Yuille [138] introduced a segmentation approach called *Region Competition* which unifies aspects of 'snake/balloon' algorithms and region-growing. The algorithm is derived by minimising a generalised *minimum description length* criterion which is a global energy function given in Equation (2.16).

$$E(\Gamma, \{\alpha_i\}) = \sum_{i=1}^{M} \left(\frac{\xi}{2} \int_{\Gamma_i} ds - \int_{R_i} \int_{w_{(x,y)}} \log P(I_{(u,v)} | \mu_i, \sigma_i) du dv dx dy + \lambda \right) \quad (2.16)$$

where M is the number of regions contained in the image, $P(I_{(u,v)}|\mu_i, \sigma_i)$ is the Gaussian density function of the intensity $I_{(u,v)}$ at pixel (u, v) with mean μ_i and variance σ_i corresponding to region R_i, Γ_i is the boundary of

region R_i, $w_{(x,y)}$ is a circular window of size n centred at pixel (x, y), (u, v) is in $w_{(x,y)}$, and λ is the code length needed to describe the density and code system for region R_i. The first term within the parentheses of Equation (2.16) is the length of region boundary Γ_i and the second term is the cost for coding the intensity of every pixel inside the corresponding region according to the probability density function.

Their algorithm starts with a random initial segmentation by putting N seeds randomly across the image. All background which is not occupied by any seed regions is treated as a single region with uniform probability distribution. To minimise the energy function, an iterative operation consisting of two steps is conducted until the energy reaches a minimum. In the first step, they fix the boundary Γ, i.e., the regions are fixed, and compute the parameter set α_i by maximising $P(I|\alpha_i)$. Secondly, α_i is fixed, and the boundary Γ_i is adjusted at each boundary pixel in the direction of the norm under a combination of a smoothing force and a static one. The smoothing force tries to make the boundary as straight as possible. The static force comes into play only when two seed regions meet and the two regions commence a competition to attract the pixel. Once the energy converges, the background region is assigned a new seed and the procedure of minimising the energy function resumes until the background region is completely occupied by seed regions. The algorithm then undergoes a region merging operation in which two adjacent regions are merged if the merging causes the largest energy decrease. Each region merging causes a new round of region competition. The competition algorithm ends when no merge can decrease the energy any further. Application of this algorithm was performed on a textured image using two texture filters and the result showed that, although the algorithm

demonstrates some interesting features, large covariance change between regions makes a bias of the boundary position noticeable and the similarity of the covariance matrices of different regions due to uneven illumination or shadows will downgrade the segmentation quality.

Bhalerao and Wilson [7] put forward a multiresolution approach combining region and boundary processes, with a region adjacency graph and a boundary adjacency graph respectively. In the region adjacency graph, each node is iteratively averaged with its neighbours and 'links' connecting nodes are switched 'on' or 'off' depending on the mean differences between them. In the boundary adjacency graph, an estimation of the gradient is performed on each node and the magnitude of the estimated gradient is used as a measure of the local edge energy and the argument as the orientation. The orientation estimate is then used to control the relaxation of the boundary graph. A 'dual' of the region adjacency graph is created by linking together the boundary nodes. As the iterations proceed, pertinent information in the boundary process can be used to augment the region-linking decision and the existence of an 'on' link in the region process tends to cause intersecting boundary links to be turned 'off'. The performance of the algorithm is excellent when applied to some synthetic images, but the segmentation is not satisfactory on natural images because the underlying image model did not take sufficient account of luminance variation across regions.

Ahuja introduced a transform for integrated detection of image edges and regions for general purpose and low-level image segmentation [3]. The main feature of the transform is that it allows pixels to group recursively to form regions in a bottom-up manner, instead of fitting models of region geometry and intensity. The transform converts an image into a force field

by computing a force at each pixel which represents the pixel's affinity to the rest of the image. The force vector at each pixel is a function of a pair of scale parameters (σ_s, σ_g). σ_s is the spatial scale parameter which denotes the shortest distance of a pixel to the region boundary and thus carries the region boundary information. σ_g is the photometric scale which denotes the intensity contrast of region with the surrounding and thus conveys the region interior information. The force field captures distinct signatures of image regions which reflect the distributed evidence of the presence of a region. The detection of regions is achieved by locating contours of force divergence. However, no specific algorithm for using the transformation to automatically estimate the scales and identify region signatures was given. Also the method was not tested on textured images although the author claimed its applicability.

Haddon and Boyce [43] proposed a two-stage segmentation algorithm which estimates the initial segmentation based on the location of the intensities of each pixel and its neighbours within the co-occurrence matrix and a relaxation labelling is employed at the second stage to impose local consistency. At the initial segmentation stage, a location in the space of the co-occurrence matrix is defined as a function of the intensities of a pixel x and its neighbour. This location in turn defines the probabilities of the pixel being interior to a region R_x and being on the boundary bordering the region R_x. At the second stage, local consistency of pixel classification is imposed by minimising the entropy of local information, where the inter-region compatibility coefficients are derived from the magnitudes of the peaks in the co-occurrence matrices and those of the boundary are given *a priori* values based on general constraints of boundary connectivity. This technique pro-

vided a simultaneous solution of the region and edge segmentation problem using global information in a local context which is able to adapt to different image characteristics. However, the performance of this approach is limited if the clusters in co-occurrence space have substantial overlaps.

2.3 Summary

Several papers which are representative of general feature extraction and segmentation methods of different categories have been selected for review in this chapter. Their advantages and limitations were also discussed. However, it is difficult to compare their performance quantitatively because there is no accepted standard set of textures on which to base a performance measure. Nevertheless, in the applications of feature extraction, the survey conducted above indicates that it is clearly preferable to have a set of feature descriptors which is able to capture both statistical and structural features of textures. Methods designed to detect statistical or structural features alone are bound to miss out the features of the other type. In the first part of Chapter 4, we will present a two-component texture model in which one component is an affine deformation, representing the structural or deterministic elements and the other is a stochastic one based on the local Fourier energy spectrum

Furthermore, the survey on segmentation methods shows that region-based processing, which only makes use of interior statistics of regions, very often generates irregular boundaries and small holes, while boundary-based processing, which only makes use of information along the boundary, does not guarantee closed boundary contours. Therefore, it is natural to integrate both region and boundary-based processes, by incorporating the duality of regional and boundary information to take advantages of one approach to

complement the other. However, the reported work of this nature is limited so far, and because of the difficulties in combining the two processes and detecting boundary information between textured regions, most of the few reported works are not capable of segmenting textures. It is this reason which motivates the work presented in Chapter 5.

Moreover, some texture features appearing at a specific scale may not show up at other scales. Single scale methods are likely to miss important features which do not appear at the selected scale. Therefore, attempting to segment images at a single resolution or scale is unlikely to get satisfactory results because of the inherent class-position uncertainty. Also, most segmentation methods rely on some training stage or ad hoc decisions, such as the placing of seed points in region growing, and splitting and emerging criteria in split and merge. Unsupervised methods requiring no training or ad hoc rules are preferred. To attack the aforementioned shortcomings, we, first, propose an unsupervised multiresolution segmentation method using Markov Random Fields based on the regional features extracted using the two-component texture model in the second part of Chapter 4. Then, in Chapter 5, an approach integrating both region and boundary processes is presented.

Chapter 3

Multiresolution Markov Random Fields

3.1 Aims

Texture segmentation can be treated as an operation mapping a target image from its intensity domain into a class domain by assigning a suitable label to each pixel, based on some extracted feature vectors. If we treat a textured image as a function defined on a regular lattice of size $M \times M$ with L possible texture classes/regions, then some symbols can be specified as follows:

$\mathcal{S} = \{(i,j) \mid (i,j)$ is the coordinate of a site in the image; $1 \leq i,j \leq M\}$

$\Gamma = \{\gamma \mid \gamma$ is a class label; $1 \leq \gamma \leq L\}$

$\lambda = \{\lambda_s \mid s \in \mathcal{S}$ and $\lambda_s \in \Gamma$, is the class label of site s$\}$

$X = \{X_s \mid s \in \mathcal{S}$ and X_s is the measurement at site s$\}$

$\Lambda = \{\lambda \mid \lambda$ is the labelling configuration of the lattice$\}$

Now, in the context of texture segmentation, what we want to achieve is to assign a proper texture class label to each pixel in an image based on

the given data such as average gray level within a window centred at that pixel or local texture features extracted in the neighbourhood of the pixel. An equivalent statistical description of this problem is that we want to know $P(\lambda|X)$ based on the symbols defined above. According to Bayes' theorem, this posterior probability distribution is a combination of a prior probability $P(\lambda)$ and the conditional probability of the data, given the labels $P(X|\lambda)$, i.e.

$$P(\lambda|X) = \frac{P(X|\lambda)P(\lambda)}{P(X)} \tag{3.1}$$

where

$$P(X) = \sum_{\lambda' \in \Lambda} P(X|\lambda')P(\lambda')$$

is seen as a constant [37].

A convenient model for segmentation is that measurements at neighbouring sites in the same region are likely to be more nearly equal than those at neighbours belonging to different regions. This suggests that we use differences in measurements as the data. Denoting $X_{ss'} = |X_s - X_{s'}|^2$ as the squared difference between the measurements at sites s and s' and $\mathcal{C}_2$ as the second order clique system (see Section 3.2.1), then $P(X|\lambda)$ may be expressed as

$$\begin{aligned} P(X|\lambda) &= \prod_{(s,s') \in \mathcal{C}_2} P(X_{ss'}|\lambda_s, \lambda_{s'}) \\ &= \prod_{\substack{(s,s') \in \mathcal{C}_2, \\ \lambda_s = \lambda_{s'}}} P(X_{ss'}|\lambda_s = \lambda_{s'}) \prod_{\substack{(s,s') \in \mathcal{C}_2, \\ \lambda_s \neq \lambda_{s'}}} P(X_{ss'}|\lambda_s \neq \lambda_{s'}) \end{aligned} \tag{3.2}$$

where $P(X_{ss'}|\lambda_s = \lambda_{s'})$ ($P(X_{ss'}|\lambda_s \neq \lambda_{s'})$) is the probability of $X_{ss'}$ given that site s and s' belong to the same (different) region and assumed Gaussian as will be discussed in Section 4.2.1,

In its application to texture segmentation, Bayes' theorem can be used to calculate the *Maximum A Posteriori* (MAP) estimate, i.e. the mode of $P(\lambda|X)$, so as to find the optimal class label configuration for the image. Knowing the properties of $P(X_{ss'}|\lambda_s = \lambda_{s'})$ and $P(X_{ss'}|\lambda_s \neq \lambda_{s'})$, the task of maximising $P(\lambda|X)$ can be achieved by maximising $P(\lambda)$. However, the computational cost of the optimisation is far beyond acceptable bounds because there are $L^{M\times M}$ configurations to be taken into account.. Instead of exhausting all the configurations, some simplifications have to be employed. Since texture is not a pixel property but a regional one, when texture is dealt with, the interaction between a pixel and its neighbourhood must be involved. This local characteristic makes Markov Random Fields a good candidate for achieving our segmentation objective at a relatively low computational cost.

3.2 Markov Random Fields

A Markov Random Field is a family of random variables $\lambda = \{\lambda_s|s \in \mathcal{S}\}$ with respect to a neighbourhood system $\mathcal{N}_s$, in which each random variable λ_s takes a value in Γ, if and only if the following two properties are satisfied:

$$P(\lambda) > 0 \qquad \forall\, \lambda \in \Lambda \tag{3.3}$$

$$P(\lambda_s|\lambda_{\mathcal{S}-\{s\}}) = P(\lambda_s|\lambda_{\mathcal{N}_s}) \tag{3.4}$$

where $\lambda_{\mathcal{S}-\{s\}}$ is the configuration of the set $\mathcal{S} - \{s\}$ and $\lambda_{\mathcal{N}_s}$ is the configuration of neighbourhood of site s, $\mathcal{N}_s$. Besag [6] reasoned that if Equation (3.3) is satisfied, the joint probability $P(\lambda)$ of any random field is uniquely determined by its local conditional probability. Equation (3.4) is the local characteristic of Markov Random Field, called *Markovianity* and suggests that a Markov Random Field is a conditional probability measure of random

variables at a site depending only on the interactions with its neighbours within a neighbourhood $\mathcal{N}_s$ [37]. In the image analysis community, this local characteristic implies that the statistical structure of the image is essentially localised within the neighbourhood system.

3.2.1 Neighbourhoods and Clique Systems

To segment regions of homogeneous texture, we need to connect measurement sites which have similar properties. This can be done by defining a neighbourhood system

$$\mathcal{N} = \{\mathcal{N}_s | \, \forall s \in \mathcal{S}\} \tag{3.5}$$

where $\mathcal{N}_s$ is the sub-set of $\mathcal{S}$ consisting of the sites neighbouring site s in some sense. The neighbour relation $\mathcal{R}$ over $\mathcal{N}_s$ is:

- *non-reflexive* : for any s, $s \, \mathcal{R} \, s$ is not true, i.e. a site is not a neighbour to itself.

- *symmetric* : for any two sites a and b in $\mathcal{N}_s$, if $a \, \mathcal{R} \, b$, then $b \, \mathcal{R} \, a$.

For a two dimensional image lattice $\mathcal{S}$, $\mathcal{N}_s$ usually covers the sites within a radius of r centred at site s but excluding s:

$$\mathcal{N}_s = \{s' | s, s' \in \mathcal{S}, \|s - s'\|^2 \leq r, \text{ and } s \neq s'\} \tag{3.6}$$

where $\|s - s'\|^2$ is the Euclidean distance between site s and s' and r is a natural number. The sites at or near the image borders have fewer neighbours than the inner sites depending on radius r. The simplest widely adopted neighbourhood is the so called first-order neighbourhood system, also called 4-neighbour system. The four sites numbered "1" in Figure 3.1 are the constituent members of the first-order neighbourhood system of site s. The

2	1	2
1	S	1
2	1	2

Figure 3.1: Neighbourhood systems of site s

second-order neighbourhood system, also called 8-neighbour system, of site s is {first-order neighbourhood} $\cup$ {the sites numbered "2"}.

Within a neighbourhood system, a clique system c is defined as a subset of $\mathcal{N}_s$ consisting of single site, *singleton*, $\mathcal{C}_1 = \{s\}$, and/or a pair of neighbouring sites, *doubleton*, $\mathcal{C}_2 = \{s, s'\}$, and /or a triple of neighbouring sites, *tripleton* $\mathcal{C}_3 = \{s, s', s''\}$, and so on. Thus $\mathcal{C}$, the family of all cliques c's is

$$\mathcal{C} = \mathcal{C}_1 \cup \mathcal{C}_2 \cup \mathcal{C}_3 \cup \cdots \tag{3.7}$$

where

$$\mathcal{C}_1 = \{s | s \in \mathcal{S}\} \tag{3.8}$$

$$\mathcal{C}_2 = \{\{s, s'\} | \text{s} \in \mathcal{S} \text{ and s' } \in \mathcal{N}_s\} \tag{3.9}$$

$$\begin{aligned} \mathcal{C}_3 \quad = \{\{s, s', s''\} | \quad & s \in \mathcal{S} \text{ and s', s" } \in \mathcal{N}_s; \\ & \text{s, s', s" are neighbours to one another}\} \end{aligned} \tag{3.10}$$

Rectangle R_1 in Figure 3.2 encompasses all the three possible clique types of a first-order neighbourhood system defined on a regular lattice $\mathcal{S}$ and rectangle R_2 includes all the possible clique types of a second-order neighbourhood

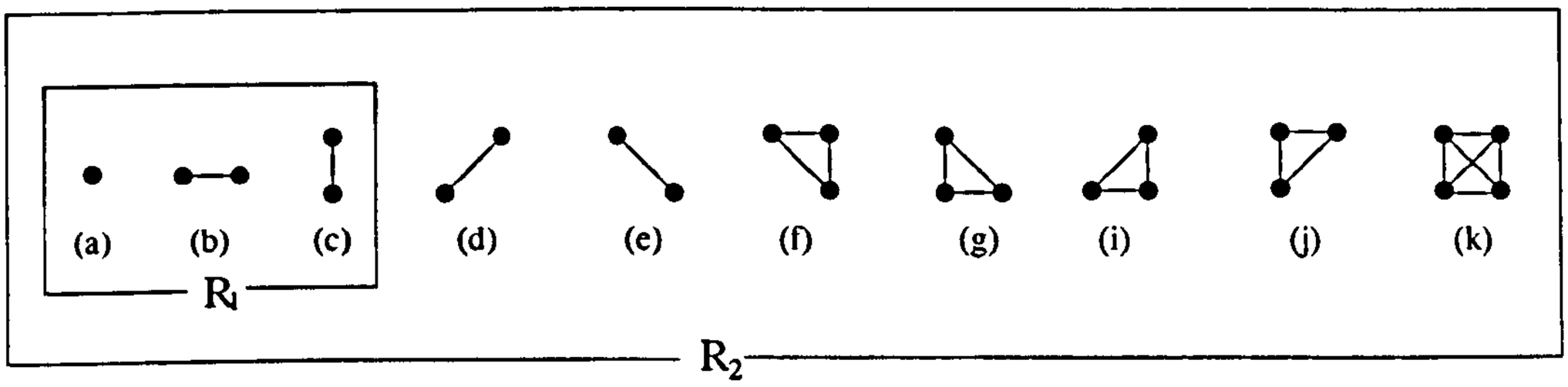


Figure 3.2: Cliques within the first-order and the second-order neighbourhood systems of site s.

system. Since the segmentation is based on difference measurements, it is appropriate to use $\mathcal{C}_2$.

3.2.2 MRF-Gibbs Equivalence

Although the local characteristic makes MRF models attractive, some researchers [6][17] pointed out the following disadvantages of an MRF:

- The joint probability $P(\lambda)$ can not be deduced without elaboration from the conditional probability $P(\lambda_s|\lambda_{\mathcal{N}_s})$ of the label at s, given the labels at $s' \in \mathcal{N}_s$.

- Specifying local characteristics is nearly impossible because it is extremely difficult to determine when a set of functions $f(\lambda_s|\lambda_{\mathcal{N}_s})$ are conditional probabilities $P(\lambda_s|\lambda_{\mathcal{N}_s})$ for a unique distribution on Λ, due to the non-trivial and highly restrictive consistency conditions imposed on them.

- The stable configuration of a statistical process is specified in terms of the joint probability rather than the conditional probability.

Fortunately, the Hammersley and Clifford expansion in [6] has established the theorem of MRF-Gibbs equivalence that λ is *a Markov random field on*

S with respect to $\mathcal{N}$ if and only if λ is a Gibbs random field on S with respect to $\mathcal{N}$. Readers are referred to [6][69] for the proof of the theorem. The importance of this theorem is that it provides a simple way of formulating the joint probability by specifying the clique potential functions, instead of local characteristics, appropriately chosen for desired behaviour of the system. In the context of texture segmentation, we can treat an image as a sample of Gibbs random field with respect to a neighbourhood system $\mathcal{N}$ if and only if its label configuration obeys a Gibbs distribution [78]. A Gibbs distribution is given, for a given set of measurements X, as

$$P(\lambda) = \frac{1}{Z} e^{-\frac{U(\lambda,X)}{T}} \tag{3.11}$$

where T is the *temperature* and the normalising factor Z is the *partition function* defined as

$$Z = \sum_{\lambda \in \Lambda} e^{-\frac{U(\lambda,X)}{T}} \tag{3.12}$$

To each configuration λ, an associated interaction energy $U(\lambda, X)$ is defined as

$$U(\lambda, X) = \sum_{c \in C} V_c(\lambda, X) \tag{3.13}$$

where $V_c(\lambda, X)$, called a *potential*, is the interaction among the sites in clique c. In the context of texture segmentation, the higher the potential is, the more repulsive the sites within the clique are against each other, i.e. it is more likely that they belong to different classes, and vice versa.

While a Gibbs random field is characterised by its global property, the Gibbs distribution, a MRF model is characterised by its local characteristic in Equation (3.4). With the theorem of MRF-Gibbs equivalence, a MRF

model is specified as

$$P(\lambda_s|\lambda_{\mathcal{N}_s}) = \frac{1}{Z_s} e^{-\frac{U(\lambda_s,\lambda_{\mathcal{N}_s},X_{\mathcal{N}_s})}{T}} \tag{3.14}$$

where the *partition function* Z_s is defined as

$$Z_s = \sum_{\lambda_s \in \Gamma} e^{-\frac{U(\lambda_s,\lambda_{\mathcal{N}_s},X_{\mathcal{N}_s})}{T}} \tag{3.15}$$

and

$$U(\lambda_s, \lambda_{\mathcal{N}_s}, X_{\mathcal{N}_s}) = \sum_{c \in C} V_c(\lambda_s, \lambda_{\mathcal{N}_s}, X_{\mathcal{N}_s}) \tag{3.16}$$

where $X_{\mathcal{N}_s}$ represents the measurements (observed data) at the sites within the neighbourhood $\mathcal{N}_s$. Equation (3.14) suggests that the estimate of the class label at site s, λ_s, is determined locally through the interaction between site s and its neighbourhood $\mathcal{N}_s$.

MRF models are often used in conjunction with statistical estimation, such as stochastic or deterministic relaxation, so as to formulate objective functions in terms of established optimisation principles, e.g. MAP. Relaxation is an iterative process for reducing the ambiguity of labelling by minimising an energy function in which contextual constraints are encoded [106]. MAP is one of the most often adopted statistical criteria for optimisation and in fact, has been the most popular choice in MRF texture analysis. In the MRF-MAP framework using relaxation, the task of partitioning a textured image is to maximise Equation (3.1), the *a posteriori* probability. That is to say that segmentation algorithms are seeking the configuration associated with lowest energy/cost. Equation (3.14) also suggests that the distribution function depends on temperature T. At low temperatures, the posterior distribution concentrates on the label which associates with the lowest interaction energy $U(\lambda_s, \lambda_{\mathcal{N}_s}, X_{\mathcal{N}_s})$, whereas at high temperature the posterior

distribution is essentially flat and no specific label is particularly favoured. For the extreme cases, if $T = 0$, the configuration with highest probability (the mode) is always chosen and if $T = \infty$, Equation (3.14) represents an exactly uniform distribution on Λ and the label updating is purely random.

Deterministic relaxation algorithms for MAP estimates always choose the configurations associated with lowest local energy, i.e. they only allow changes from higher energy configurations to lower energy configurations. This is done by simply setting $T = 0$. On the other hand, while changes from higher energy states (lower probability) to lower energy states (higher probability) are encouraged, stochastic relaxation algorithms for MAP estimation also allow changes from lower energy states to higher energy states. This feature gives the algorithm a chance to 'climb' out of local minima and thus is adopted in the rest of this work. Stochastic relaxation in a sense of simulated annealing is carried out by reducing the temperature T iteratively. According to Equation (3.11), as the temperature T decreases at each iteration, the probabilities of the configurations associated with lower energy become larger and those with higher energy are reduced. This iterative refinement, which continues as labelling ambiguity is reduced based on the local context, by successively reducing temperature, is analogous to a physical annealing process, in which a system is 'guided' to its low energy, purified state. Eventually (hopefully), the system will settle in the state/configuration with lowest energy. Geman and Geman [37] suggested

$$T = \frac{C}{\log(1+i)} \tag{3.17}$$

as the annealing schedule, where i is the number of iterations/full sweeps of all sites, C is a sufficiently large constant and T is the temperature function of i. Geman and Geman termed this sampling scheme a Gibbs sampler

[37]. Since the 'energy landscape' over a Markov random field is usually non-convex, in order to avoid local minima it is preferable to start at high temperature and as the relaxation proceeds, T is decreased gradually in the manner of physical annealing.

In addition to the attribute of local characteristic, the Gibbs sampler has the property of ergodicity, i..e. the final configuration when the sampling converges is independent of the initial configuration when the sampling starts [37]. This means that we can initialise the labels of all the sites randomly.

3.3 Issues about Applying MRF's

Despite the advantages of Markov Random Fields, there are still some issues which call for attention:

1. From Equation (3.17), we can see that the temperature T is proportional to the value of constant C. If C is too large, the convergence rate becomes too slow to be tolerable. If C is not large enough, the temperature is always low and, as a result, the algorithm becomes more likely to settle in a configuration with minimal *local* energy instead of minimal *global* energy. Figure 3.3 shows the curves of T (as a function of the number of iterations) with respect to different values of C. Deciding the best value of C is non-trivial and the range of the interaction energy $U(\lambda_s, \lambda_{\mathcal{N}_s}, X_{\mathcal{N}_s})$ has to be taken into account so that C can be better chosen. In the multiresolution approach used in our work, since the coarser segmentation results at higher levels are propagated down to lower levels, taking a lower value for constant C at higher resolutions will accelerate the convergence rate without degrading the segmentation result at the new level.

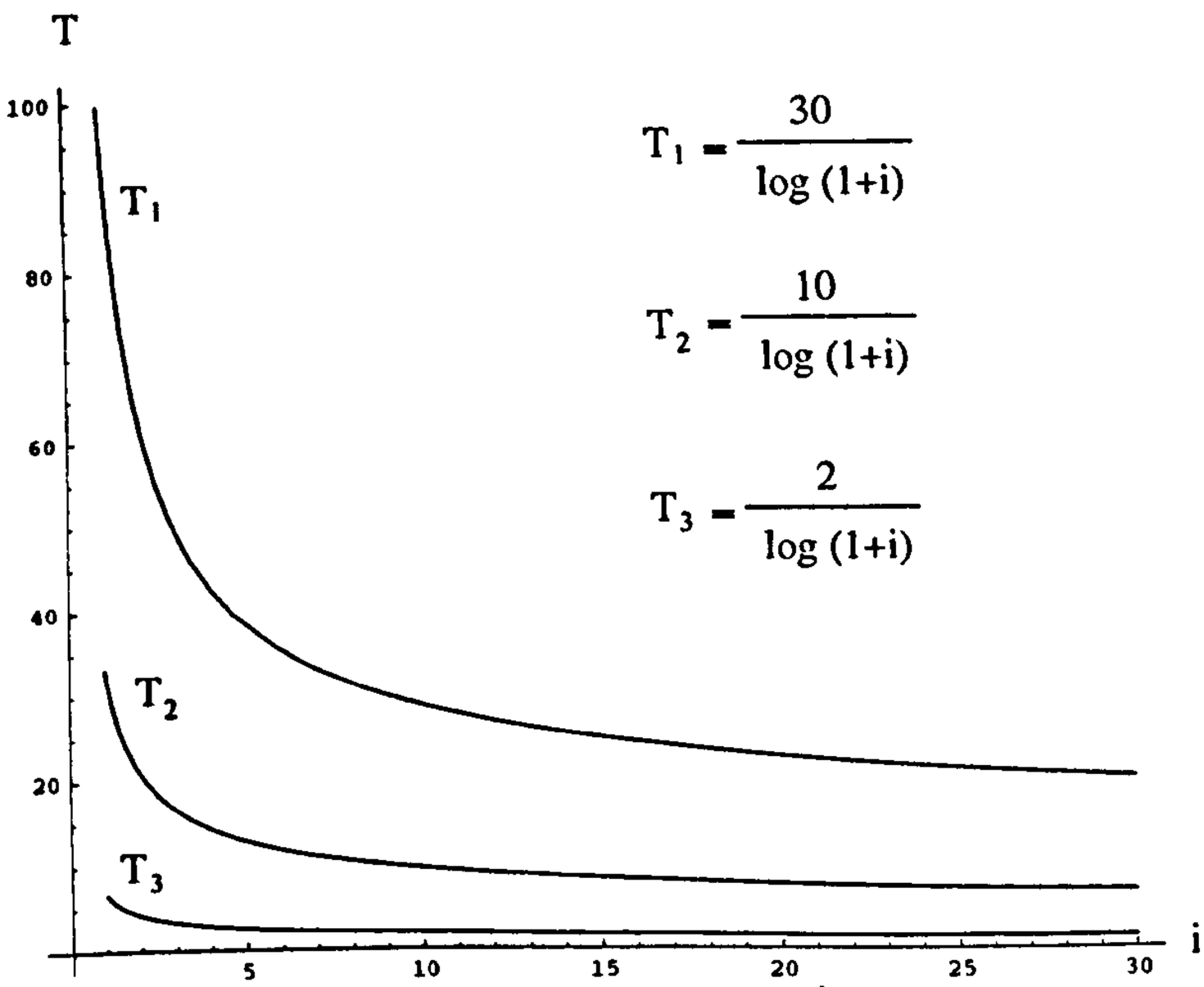


Figure 3.3: Stochastic annealing schedules.

2. For many cases, the number of texture regions in an image is not known *a priori*. If we want to make the algorithm unsupervised, it should be able to estimate the number of texture regions or classes automatically. Otherwise the algorithm must be provided with the number of texture classes. Suppose in the worst case that the algorithm has to work under supervision by requiring the number of texture classes to be specified, then from the partition function in Equation (3.15), we know that the posterior probability has to be calculated for all L texture class labels in Γ. From Equation (3.14), it is easy to see that the larger L is, the smaller the probability of each label to be selected. In this case, even for the label with lowest interaction energy (cost), its opportunity of being selected is slim because the posteriori probability of it is overwhelmed by the big population of labels. This means that if L is large, the algorithm takes more iterations or longer time before the optimal configuration can gradually emerge from the rest. On the other hand if L is given a value smaller than the actual number of texture regions, the image will inevitably be under-segmented. We will attack this issue in detail in Chapter 4 and make the algorithm unsupervised.

3. Although Geman and Geman claimed that stochastic relaxation in conjunction with the annealing schedule guarantees convergence to the global minimum [36][37], it is unrealistic not to consider the time consumption of this slow schedule and specify some stopping rule for the algorithm. However, with the adoption of a stopping rule, convergence to the global minimum is no longer guaranteed, even if the simulated annealing scheme is adopted, because the 'landscape' of the interaction energies over MRF is non-convex and it is not possible to anticipate

where the local minima are and to differentiate the global minimum from local minima. In order to enable the algorithm to jump out of local minima, in our work, we allow the algorithm to continue for a few more iterations when it settles in a configuration. Note that *settling in a configuration* means there is no change to the label of any site over an iteration. This measure still does not guarantee the convergence to the global minimum because it also gives the algorithm a chance to jump out of the global minimum. However, as the temperature falls, the possibility that the algorithm jumps out from the global minimum to a configuration with higher energy approaches 0 [37]. The experiments conducted in the next section show the advantage of this measure.

4. How the labels are updated is another issue. We could do it serially (such as raster scan) using only one processor or in parallel using multiple processors. Serial updating certainly requires more time, while parallel updating requires more sophisticated hardware and communication between processors [37]. Parallel updating can be executed synchronously, updating sites simultaneously at each time step, or asynchronously, updating them one at a time. The asynchronous case, in which each processor is driven by its own clock is more natural for both brains and computers.

3.4 A Multiresolution Approach

MRF's have been used to model textures in a Bayesian framework in the past and continue to attract researchers studying various image processing problems by treating them as well-defined statistical inference problems. However,

despite the desirable attributes of the MRF framework, the assumption of Markovianity allows global information to exist and propagate only through local interactions within defined neighbourhoods. This inherent shortcoming calls for some remedy. In addition to this drawback, MRF models at single resolution do not provide a solution to the issue of *uncertainty* addressed in Chapter 1. Moreover, some texture features appearing at one scale or resolution do not necessarily appear at another. Applying MRF at a single resolution is likely to lose some important features which is essential to successful segmentation. To circumvent the aforementioned shortcomings, multiresolution approaches appear to be an intuitive and efficient extension which have the following advantages:

- Multiple resolution approaches are efficient in allowing global information (e.g. texture class) and local information (e.g. texture boundary position) to propagate smoothly across different resolutions.

- Class-position (*what-where*) uncertainty addressed in Chapter 1 is alleviated via information fusion at different resolutions which leads to localisation in both class space and boundary position.

- Texture features occurring at different resolutions or scales can be picked up within a multiresolution scheme to aid segmentation.

- By segmenting the image at coarse resolution first, at relatively low cost (because the number of sites are smaller), the finer resolutions can be conditioned to avoid local minima which are far from the global minimum and computation is minimised.

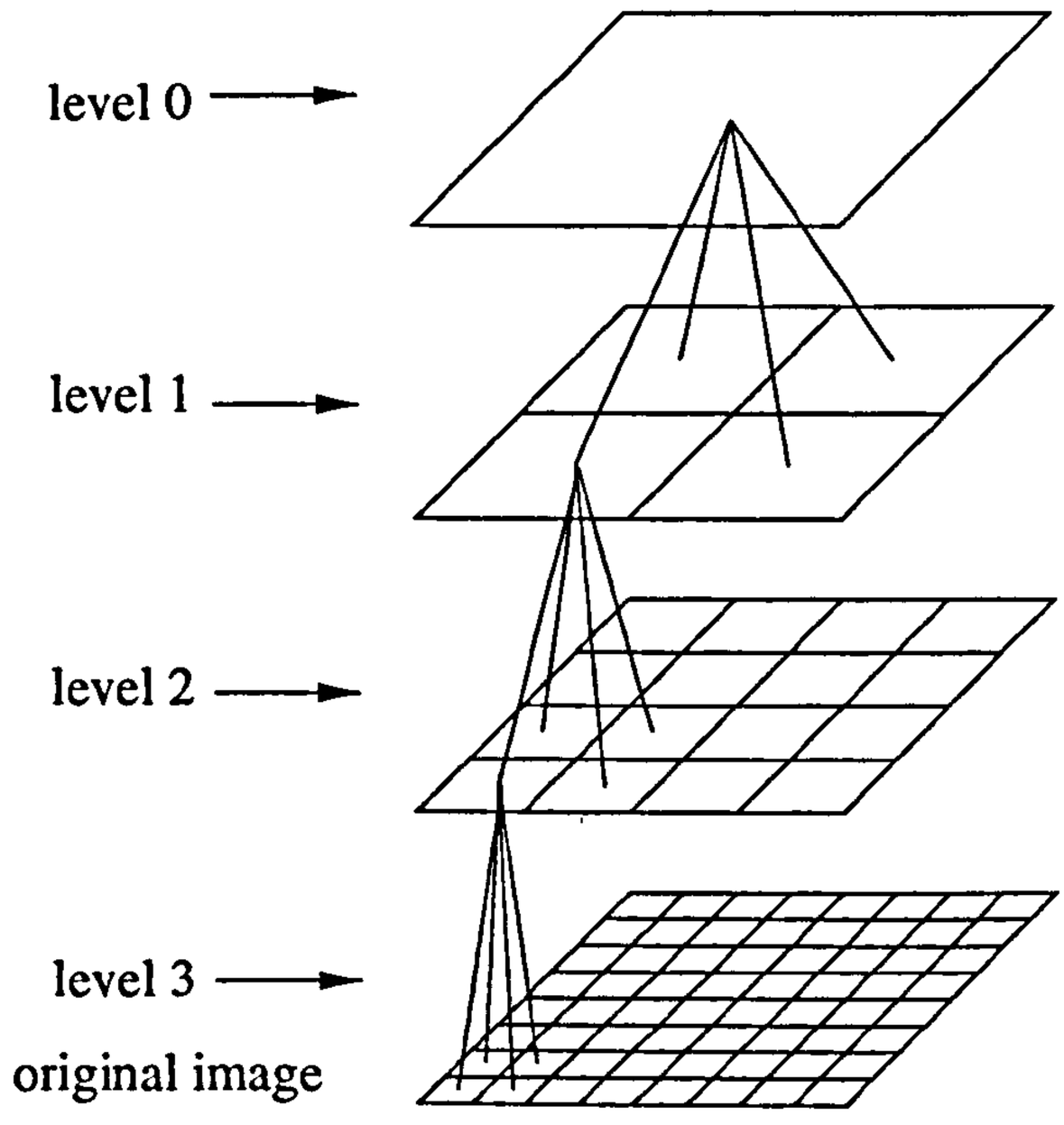


Figure 3.4: A multiresolution quadtree structure.

An example of a multiresolution pyramid conforming to a quad-tree structure is shown in Figure 3.4. Taking the original image as the bottom level (level 3 in this figure) of the structure, each higher level is created by sampling the level immediately below, such that each block at a level has four descendant blocks at that level. We can see that the higher the level (smaller level number), the higher the class resolution and the lower the position resolution. In contrast, the lower the level, the lower the class resolution and the higher the position resolution.

For such a multiresolution scheme, we can model each level of the structure as a MRF, giving a multiresolution MRF (MMRF). The segmentation algorithm using a MRF framework starts at a nominal top level other than level 0 (because it is unrealistic to start segmenting an image from a level with only one site). It is reasonable in most cases to take the level with 8×8 sites as the nominal top level. Due to *ergodicity*, the property of the sampler

that the eventual class label configuration of the image is independent of the initial configuration, at the nominal top level, the class labels of the pixels in the image are initially randomly configured. Upon convergence of the algorithm, the detected texture boundaries and class information are propagated down to next level, so as to allow refinement to be carried out at that level. Since the class resolution is higher at higher levels, to accelerate the computation, we simply take the *1 to 4* expansion of the final configuration of the immediate ancestor level of a specific level as its initial configuration. In turn, since the initial configuration at lower levels is a coarse segmentation inherited from its father level, the sampling at the new level is conditioned on the result of the previous level. Therefore, the algorithm is allowed to start from a temperature lower than the starting temperature of the previous levels. This increases the convergence rate.

In the application of MMRF framework to image segmentation, to condition the new level on the segmentation result of the previous level, we impose a neighbourhood system $\mathcal{N}_s$ consisting of the 4 first-order neighbours on the same resolution level and the *father* on the level above (the shaded sites) as shown in Figure 3.5, i.e.

$$\begin{aligned}\mathcal{N}_s = \{&(i-1,j,k),\ (i+1,j,k),(i,j-1,k),\\ &(i,j+1,k),(\lfloor i/2\rfloor,\lfloor j/2\rfloor,k-1)\}\end{aligned} \tag{3.18}$$

where (i,j,k) is the coordinates of site s at level k and $\lfloor . \rfloor$ denotes the floor of a real number. Note that at the nominal top level, since there is no ancestor level, the neighbourhood system consists of the 4 first-order neighbours only. Also note that the neighbour relation between site s and its father is not symmetric because the father site is a neighbour of s while

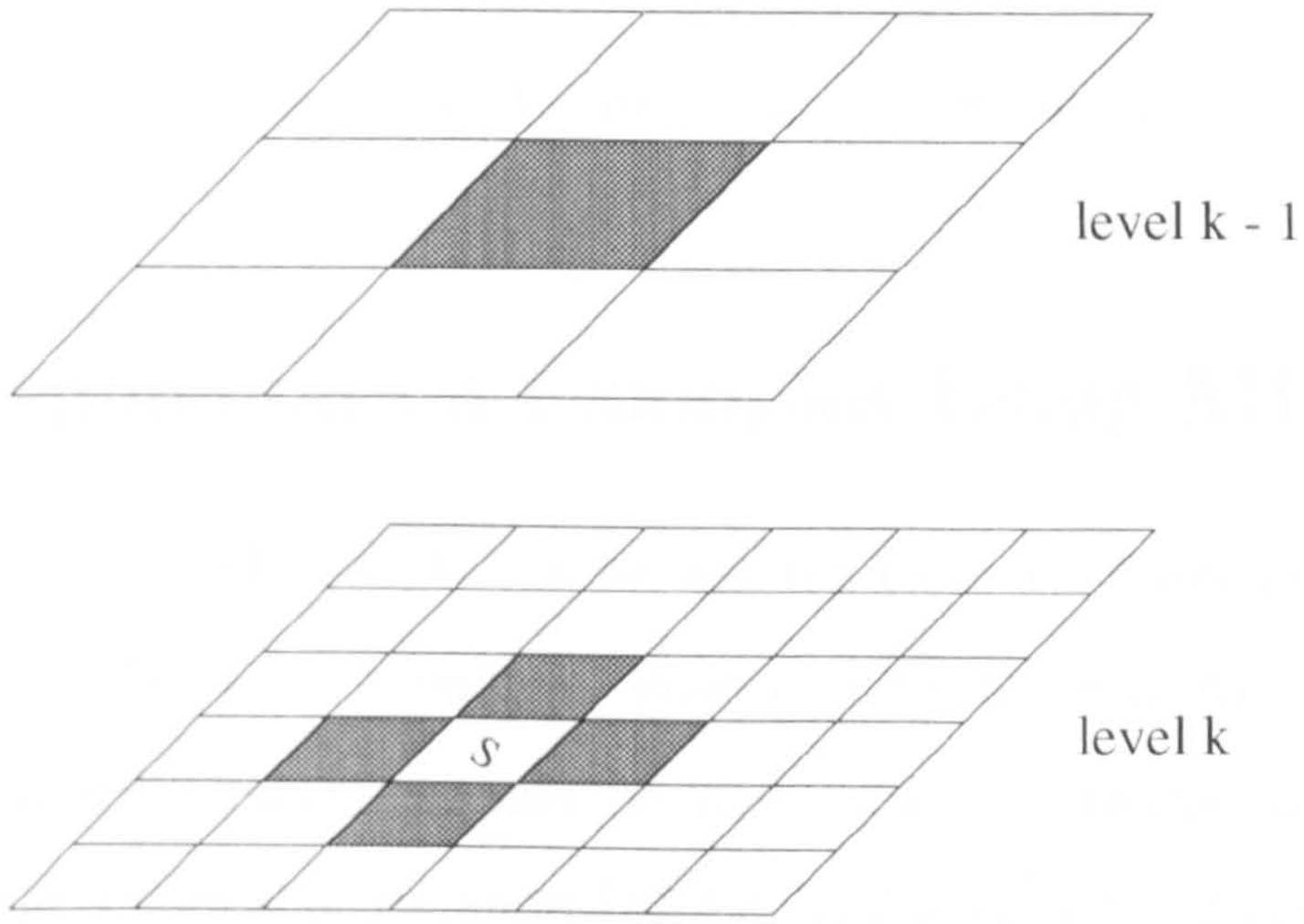


Figure 3.5: The shaded sites at different levels form the neighbourhood system imposed on the experiments of this work.

s is not a neighbour of its father site. The inclusion of the father site in the neighbourhood system is intended to condition the labelling of the children.

Conditioning the new level is achieved by allowing the information such as boundary location and the class of father site of the previous level to be propagated down to the next level and encoded in interaction energy within the neighbourhood system. This propagated information represents contextual constraints, which reflect the fact that the farther away from the boundary, the less likely is the current site to be assigned a label different from its father's.

The clique system within $\mathcal{N}_s$ employed is $\mathcal{C}_2$ only, i.e.

$$\begin{aligned} \mathcal{C} &= \mathcal{C}_2 \\ &= \{\{(i,j,k),(i-1,j,k)\},\{(i,j,k),(i+1,j,k)\}, \\ &\quad \{(i,j,k),(i,j-1,k)\},\{(i,j,k),(i,j+1,k)\}, \\ &\quad \{(i,j,k),(\lfloor i/2 \rfloor,\lfloor j/2 \rfloor,k-1)\}\} \end{aligned} \tag{3.19}$$

Equation (3.19) means that only the pair-wise interactions between site s and each of its neighbours in $\mathcal{N}_s$ are utilised to define interaction energy $U(\lambda_s, \lambda_{\mathcal{N}_s}, X_{\mathcal{N}_s})$.

3.5 Segmentation Examples Using MRF's

To show how MMRF models can be applied to image segmentation, and what factors affect their usefulness, some experiments were carried out. In the following experiments, we first build a multiresolution pyramid similar to Burt et al.'s Gaussian Pyramid [13] by employing a 4×4 Gaussian-like kernel [131]. As a measure for class discrimination, the interaction potential V_c between any site and its neighbours within the pyramid is based on the squared gray level difference, $X_{ss'} = ||X_s - X_{s'}||^2$, between them. That is to say that the only texture feature we will extract from each site is the mean gray level of that site. Since each level in the pyramid is created by convolving the Gaussian kernel with the immediate level below, the result of this local averaging operation on each site/pixel at each level is a low-pass filtered version of the level down below. So we can expect that the higher the level in the Gaussian pyramid, the lower the variance within the same regions, i.e. the higher the resolution in class space. However, the spatial resolution is reduced during the convolution and sub-sampling process.

Of cause, we can expect that using mean gray level as the only textural feature is not enough to differentiate all textures. Since the only purpose in this section is to demonstrate how MMRF is applied, a more sophisticated and robust technique, extracting more features, will be given in Chapter 4. We will also leave the detail of the calculation of $V_c(\lambda, \lambda_{\mathcal{N}_s}, X_{\mathcal{N}_s})$ until Chapter 4. In order to analyse the relationship between constant C and the

convergence rate, the number of labels allowed in the examples is fixed at four, although the algorithm to be presented in Chapter 4 does not require this number to be specified. The 128×128 pixel images used in the experiments are limited to those consisting of only two regions separated by a vertical boundary.

In summary, the segmentation algorithm employed in the following examples is :

step 1: Build Gaussian pyramid, a multiresolution gray level pyramid.

step 2: Let current level = nominal top level

step 3: Segment current level by modelling it as a MRF

step 4: *if* current level $\neq$ nominal bottom level then

propagate information gathered at current level down to next level

else go to step 6

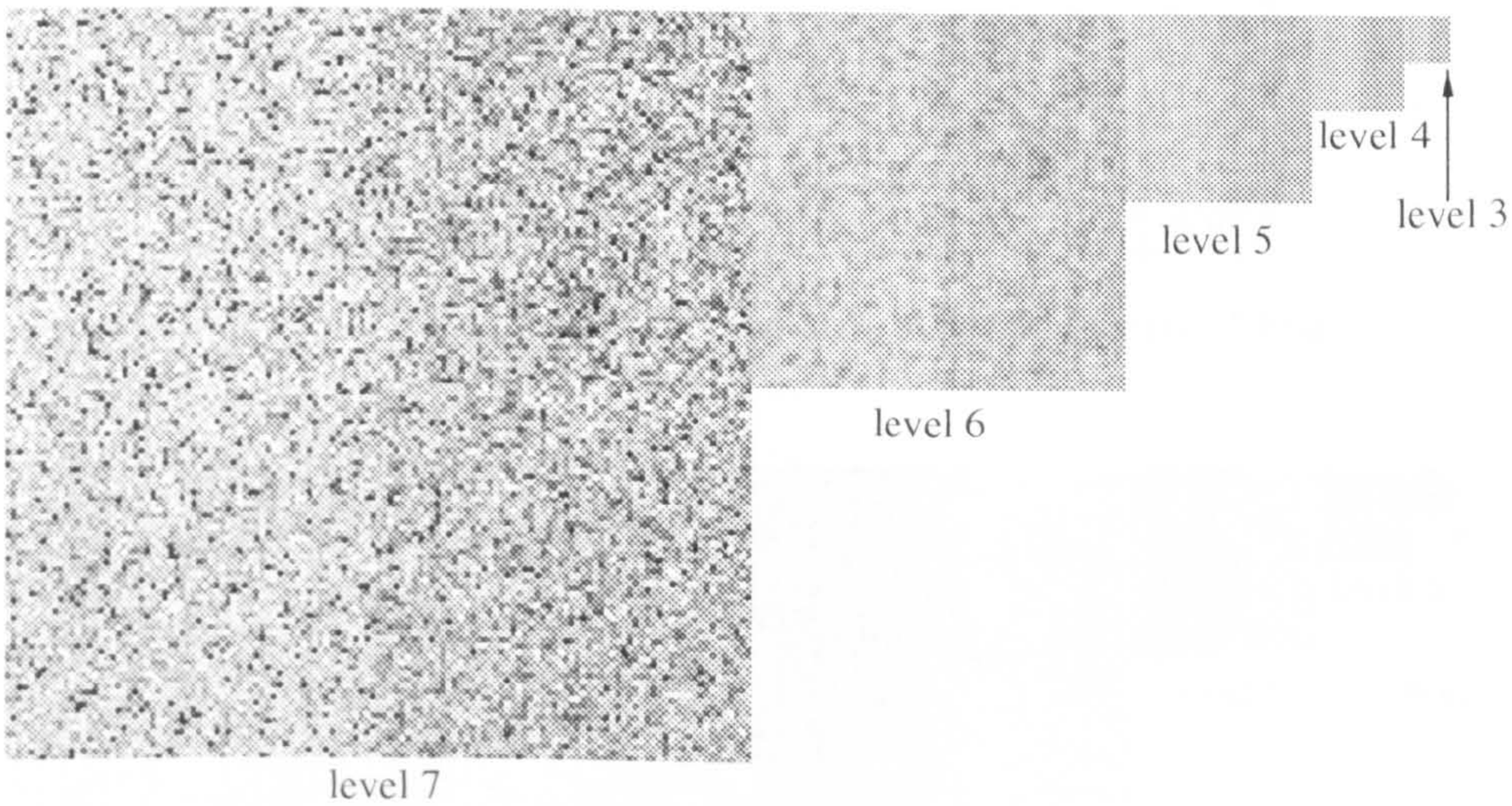
step 5: Let current level = next level, go to step 3

step 6: stop

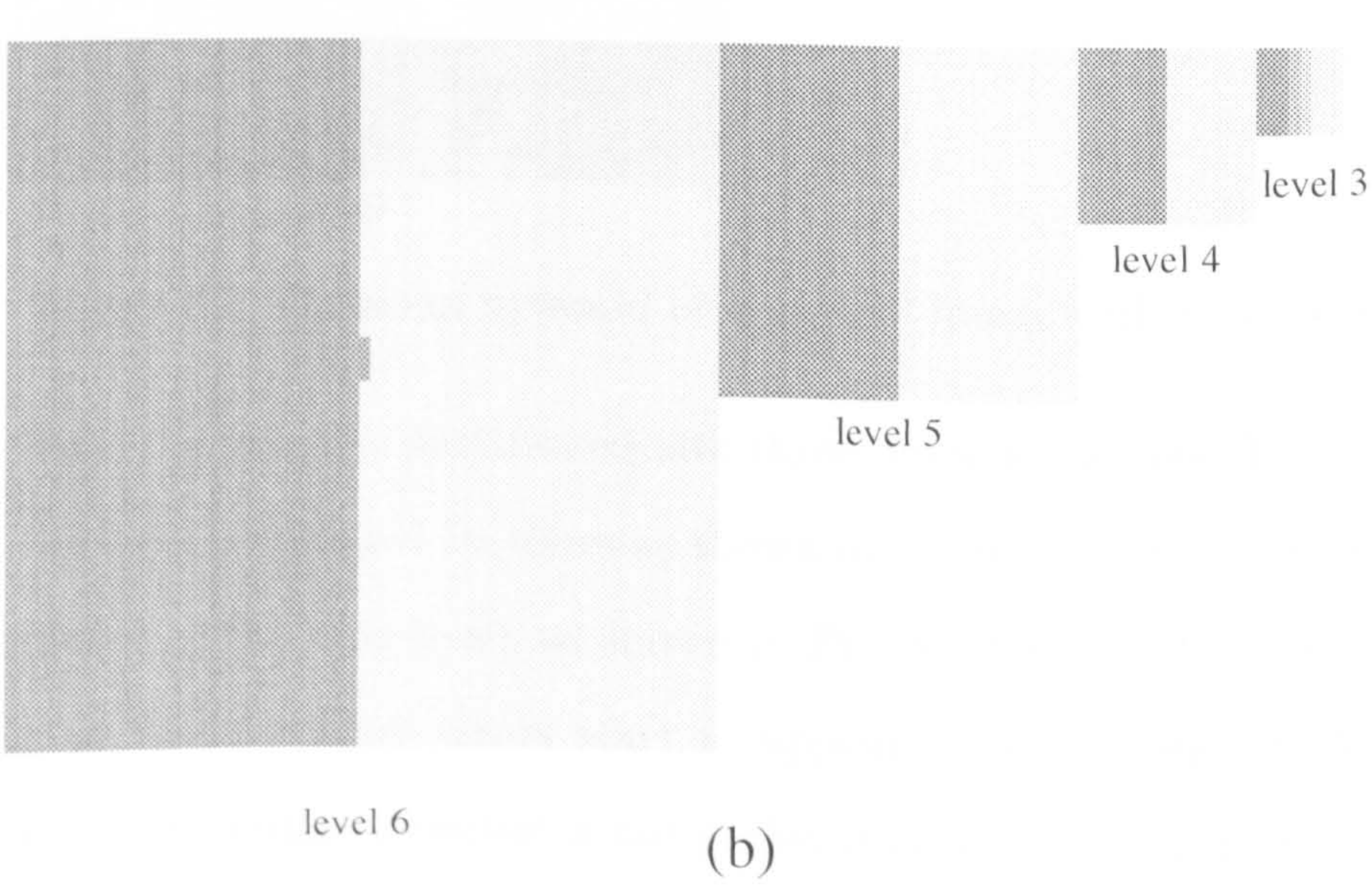
Example 1: Figure 3.6(a) is a Gaussian pyramid based on the original image (level 7) with two regions of random white noise separated by a vertical boundary right in the middle of the image. On the left-hand side of the image is the white noise with mean gray level equal to 180 and the variance equal to 900. The mean gray level of the white noise on the right-hand side of the image is 150 and the variance is 900 also. The nominal top level (level 3) consists of 8×8 blocks. Despite the squared difference between the means of the two regions being the same as the variance within each region (900),

the classification is still quite successful. Due to the lack of correlation of the noisy data at the bottom level (level 7), the algorithm stopped at level 6. Figure 3.6(b) shows the segmentation results of levels 3 down to level 6. In order to make the results clearer, every level of the segmentation result is enlarged 2 times in both dimensions. Note that the nominal top level (level 3) is partitioned into four different regions. The reason for this is that the gray levels of the sites along the physical boundary are the average values of the gray level of the sites belonging to different regions at the lower level in the Gaussian pyramid, so that they are different from the mean values of both regions. This is the nature of the averaging process and corresponds to the uncertainty of the spatial location at higher level. The gray levels of the sites at the top level are shown in Table 3.1. Although column 3 (start counting from 0) actually belongs to the left region and column 4 belongs to the right one, they are classified as two different regions. However, as the algorithm descends from the top of the pyramid, the spatial resolution becomes higher, so the actual boundary becomes more certain (see level 4 of Figure 3.6(b)) and only two regions are identified. The noise reducing effect of the Gaussian kernel is also apparent in Table 3.1. The variances (≈ 1) of the 4 partitioned regions is insignificant compared to those (900) within the two regions at the bottom level.

Example 2 : Figure 3.7 shows another image pyramid of the same size and structure as that in Figure 3.6 consisting of two regions of natural textures, burlap and sand from Brodatz's album [12]. Again, in order to show the algorithm's power of refining the boundary position, this time the vertical boundary is several pixels to the right of the middle of the image. The segmentation result at each level is shown in Figure 3.8(a). Note that at the



(a)



(b)

Figure 3.6: A noisy image and its segmentation result

180	177	177	172	153	146	148	146
178	176	178	174	155	150	149	146
178	178	178	173	154	148	148	148
179	180	178	172	155	147	147	148
179	179	176	171	155	149	150	149
178	178	177	173	157	151	150	149
178	178	178	173	156	150	149	148
179	178	179	175	154	148	148	150

Table 3.1: Gray levels of the top level of Figure 3.6(a)

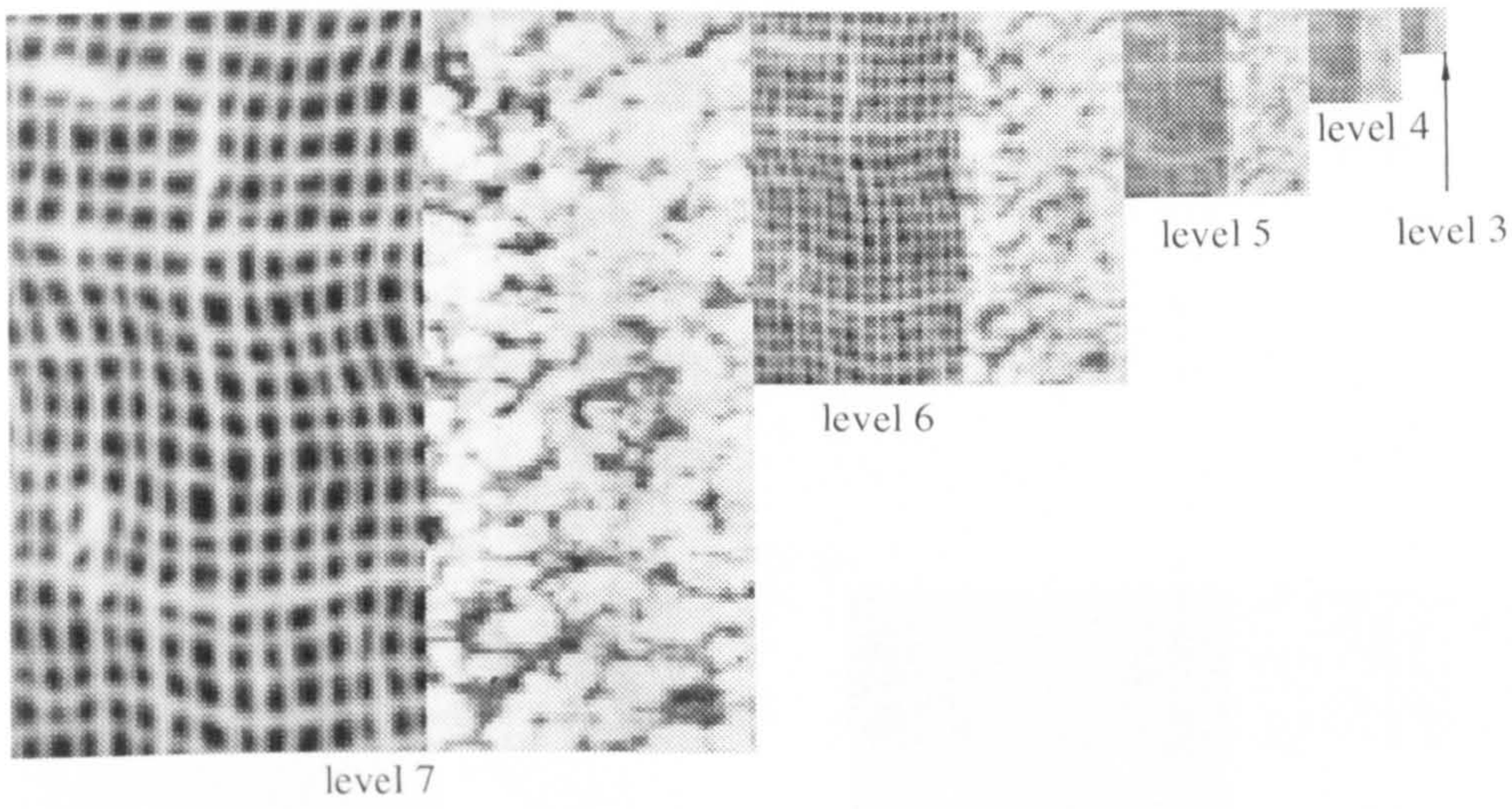
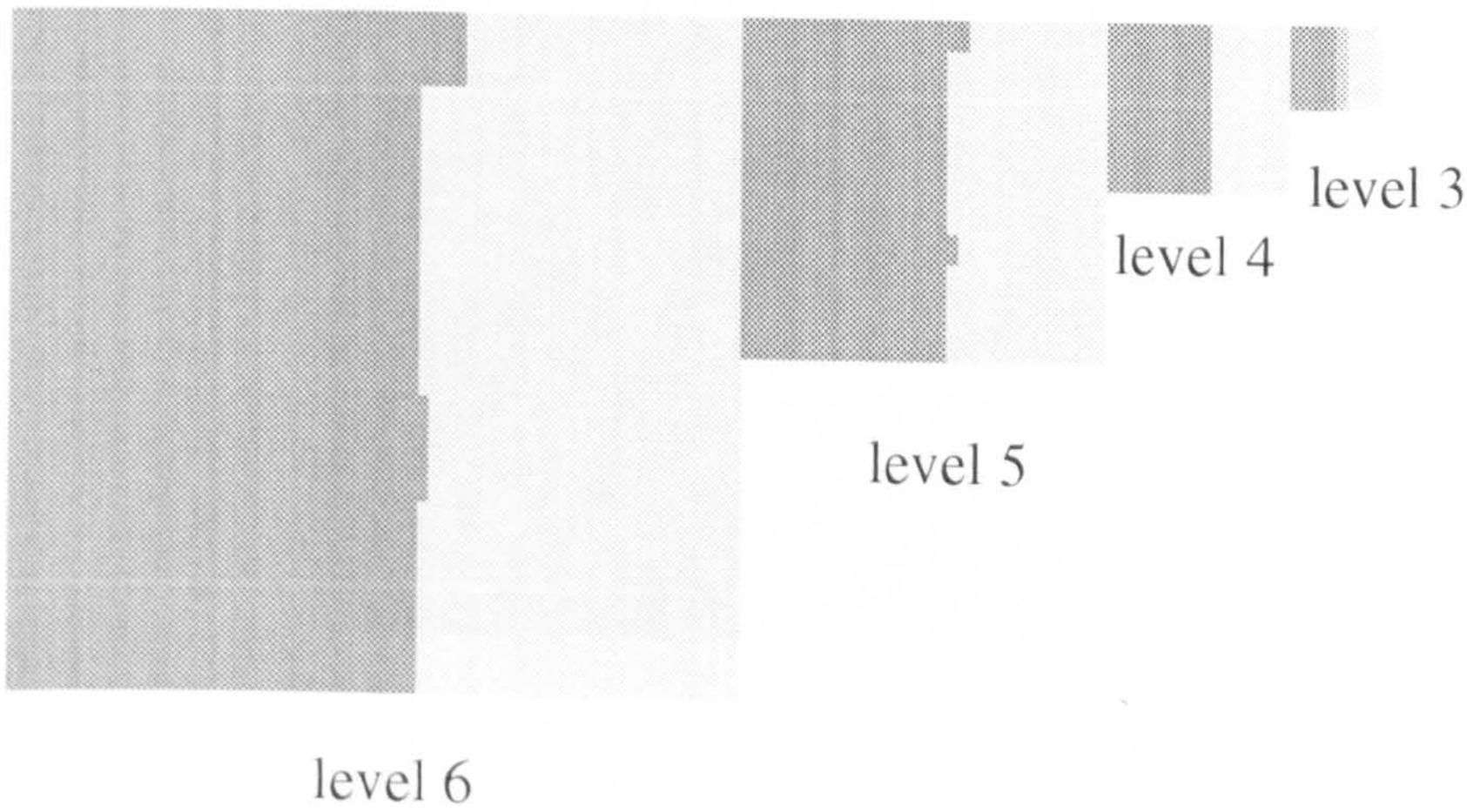


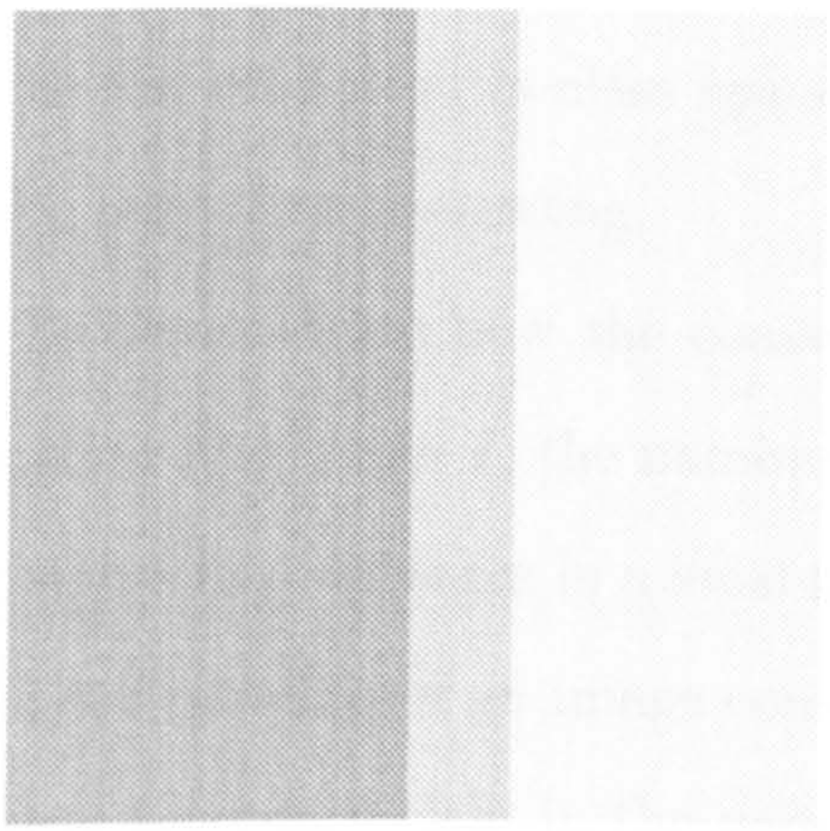
Figure 3.7: Gaussian pyramid of a natural image with two regions

top level, the image is partitioned into three different regions. The result at the top level is enlarged 16 times as shown in Figure 3.8(b). The errors are eliminated at the next level, as shown in Figure 3.8(c) (8 times enlarged). However, segmentation errors start to appear at level 4 and at the levels below (the boundary detected is not perfectly straight as it is in the original image). The reason for these errors is that the gray levels of some sites on the right-hand side of the boundary are closer to the gray levels of the sites on the opposite side of the boundary rather than the sites on the same side.

The same algorithm is also applied directly to the bottom level (level 7)

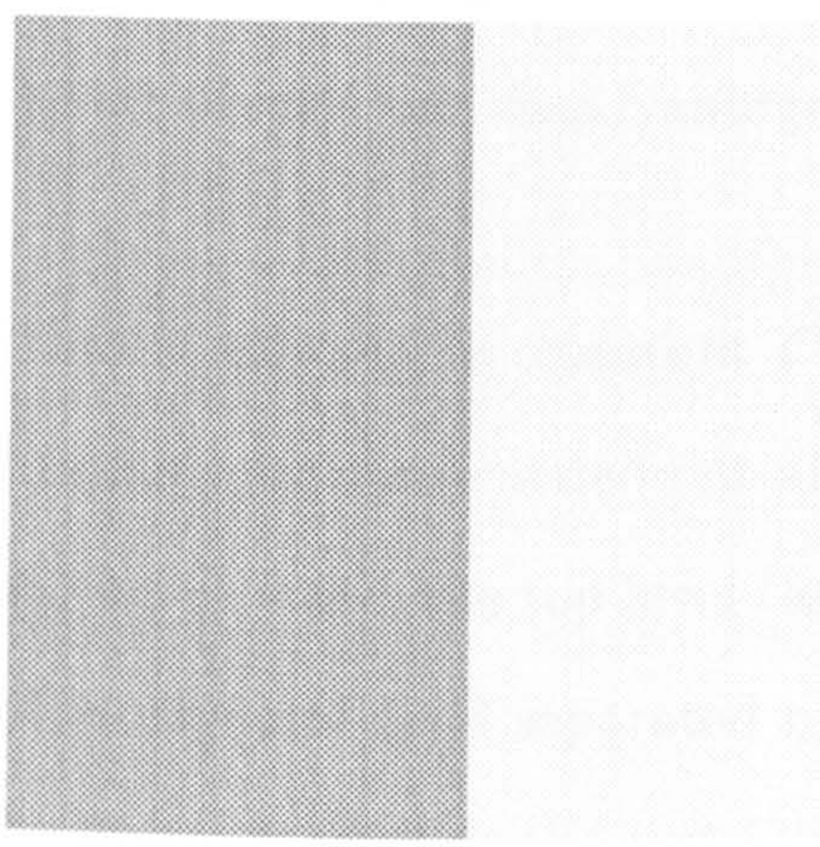


(a)



level 3 (enlarged)

(b)



level 4 (enlarged)

(c)

Figure 3.8: (a) Segmentation results of each level of Figure 3.7 (b) Enlargement of the segmentation result of level 3 (top level) (c) Enlargement of the segmentation result of level 4

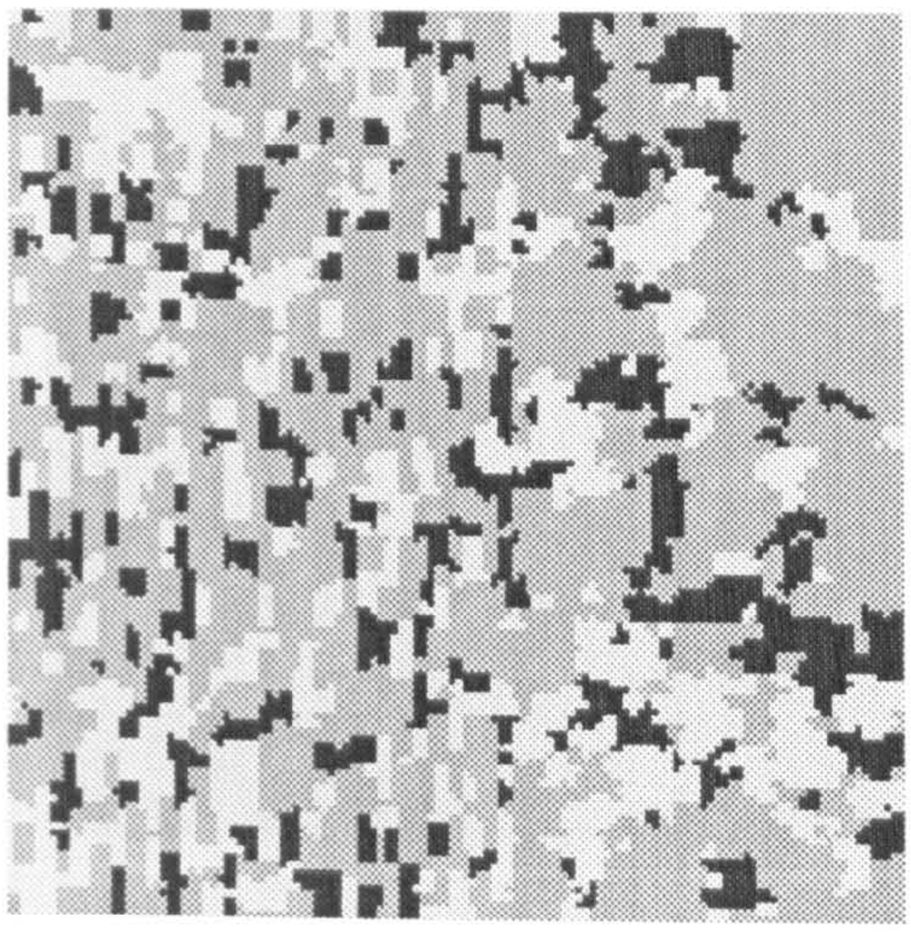


Figure 3.9: Segmentation result of applying the algorithm directly to the bottom level (level 7) of Figure 3.7.

of Figure 3.7, i.e., the original image, to get a result as shown in Figure 3.9 for comparison. Since each site at this level contains only 1 pixel, the result conforms to our expectation that the image would be over-segmented because of the low resolution in class space. This result reveals one disadvantage of single resolution processing.

To demonstrate how the convergence rate is affected by constant C of Equation (3.17) and I, the number of additional iterations carried out after the algorithm converges in a local minimum energy state, the top level (level 3) of the pyramid of an image consisting of burlap and sand separated by a vertical boundary right in the middle of the image is used as the test image (Figure 3.10, 16 times enlarged) and some data of the segmentation results are plotted in Figure 3.11 and 3.12. It is clear that when C is small, the error rate is high while the number of total iterations is small. This implies that that the algorithm is more likely to settle in a local minimum energy state rather than the global one. As C increases, which means the temperature is

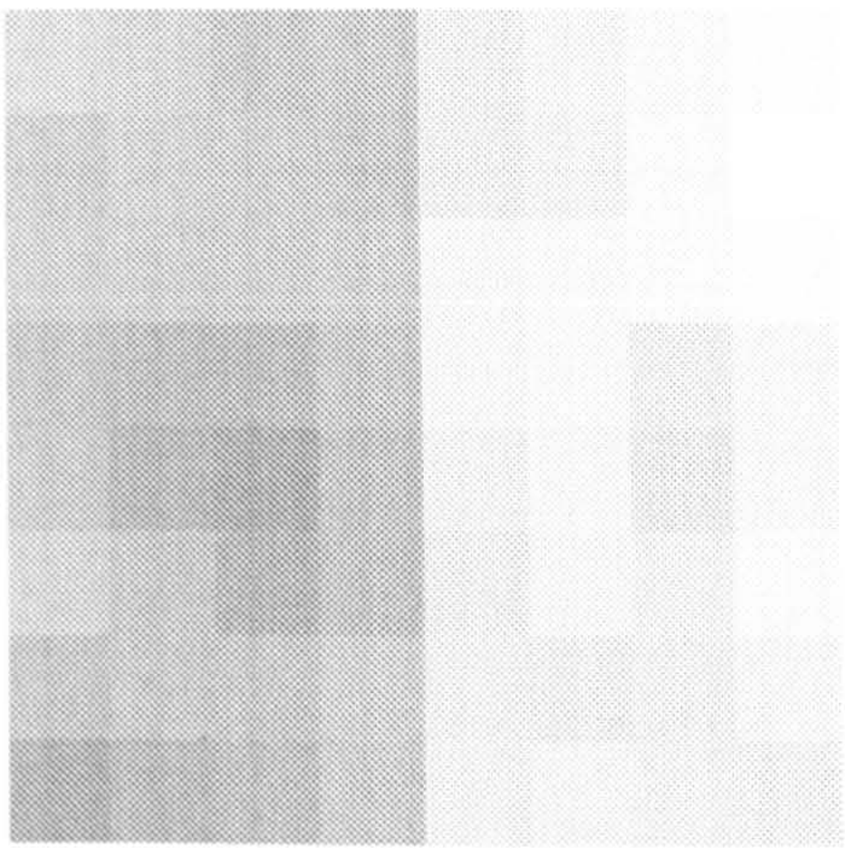


Figure 3.10: 3rd level of natural image with burlap texture on the left and sand texture on the right

higher on a given iteration, the error rate decreases correspondingly, although the total number of iterations becomes larger.

Example 3 : The algorithm is still robust even when a lot of noise is present. In order to test the robustness of the algorithm, noise is added directly to Figure 3.10, a clean 8×8 image which has been enlarged 16 times, so no noise filtering is done prior to the application of the algorithm. Figure 3.13(a) and 3.13(b) are noisy versions of Figure 3.10 with Signal to Noise Ratios (SNR) 15dB and 10dB respectively. Figure 3.14 and 3.15 show the data collected when the algorithm is applied to the clean image (Figure 3.10, SNR $= \infty$) and the noisy images (Figure 3.13). Figure 3.14 shows that the higher the SNR, i.e. the less noisy, the quicker the algorithm converges. Also as expected, the higher the SNR, the lower the error rate (see Figure 3.15).

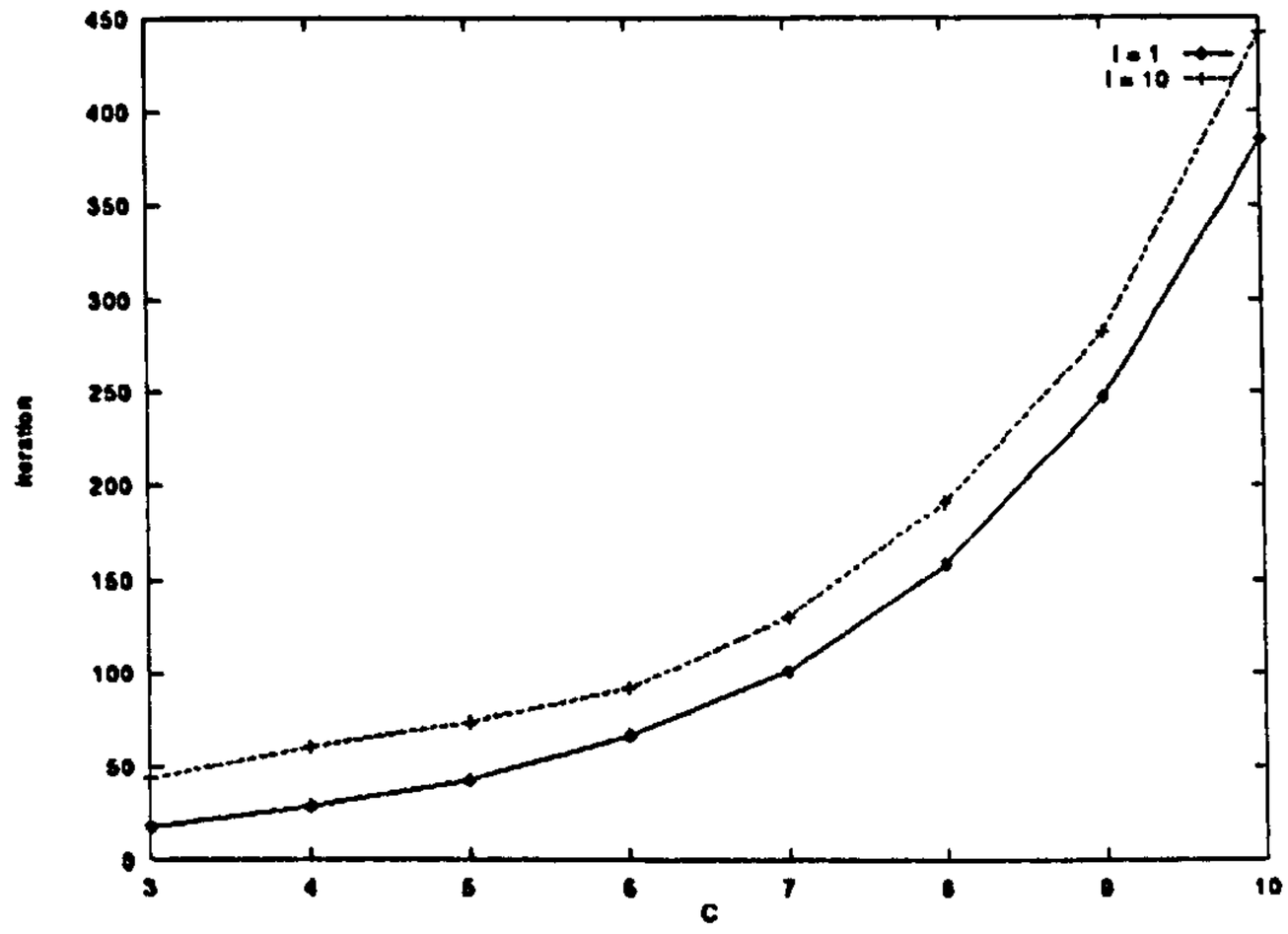


Figure 3.11: The relationship between the annealing constant C and total iterations with different I's (I is the number of additional iterations carried out after the algorithm converges in a local minimum energy state)

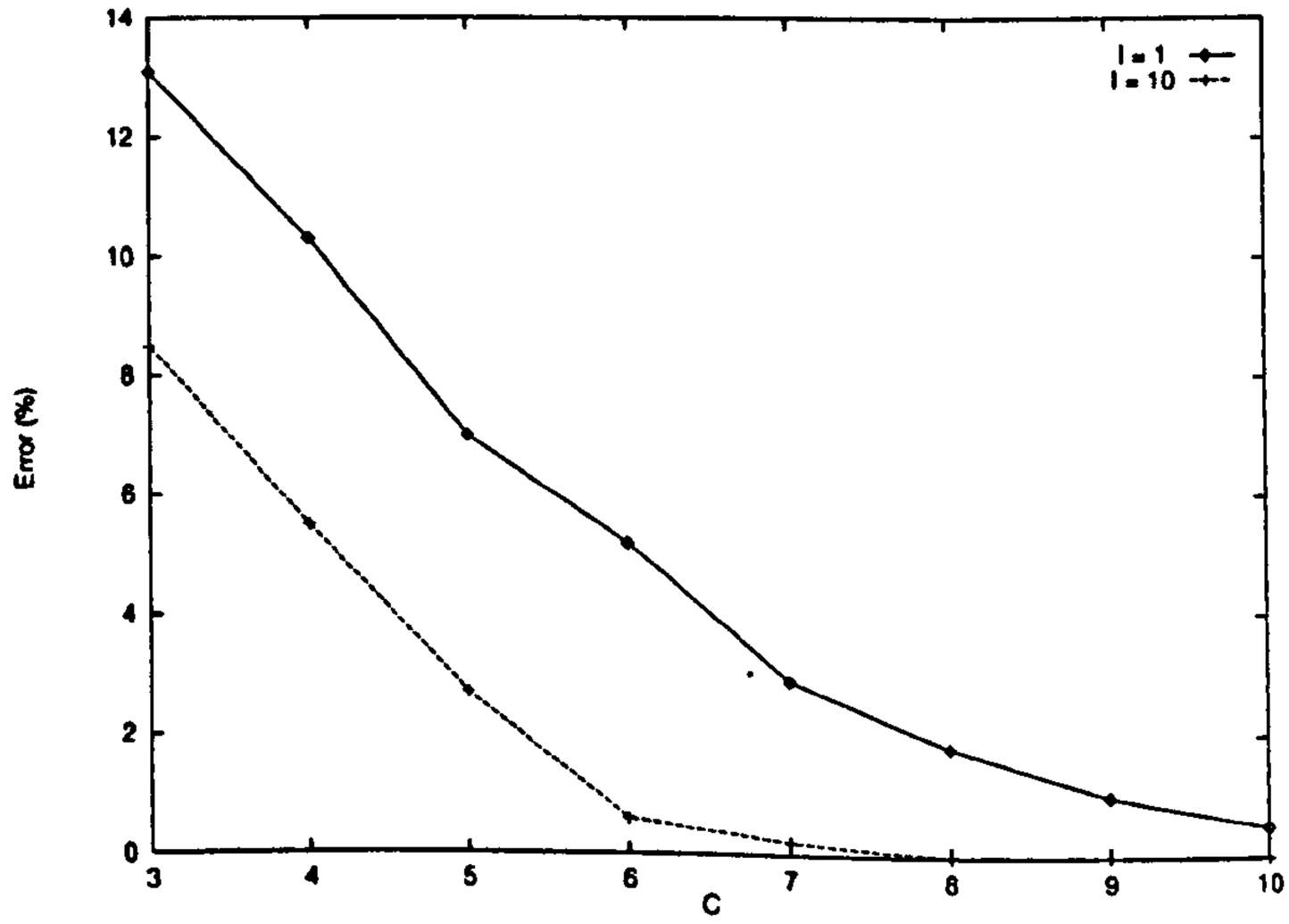
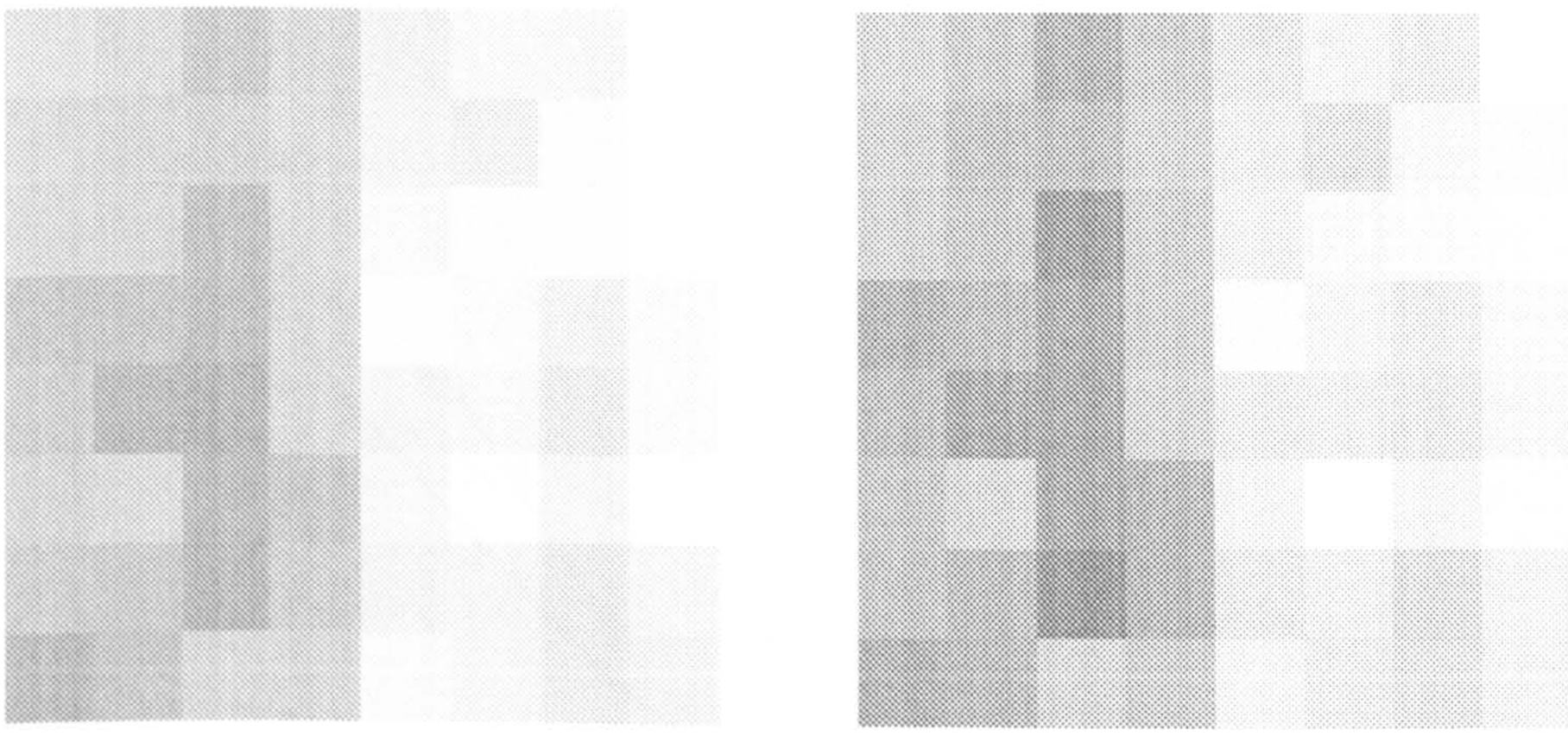


Figure 3.12: The relationship between the annealing constant C and error rate with different I's (I is the number of additional iterations carried out after the algorithm converges in a local minimum energy state)



(a) (b)

Figure 3.13: (a) Noisy version of Figure 3.10 with $SNR = 15dB$ (noise is added to Figure 3.10 directly, so no filtering is done prior to the application of the algorithm (b) Noisy version of Figure 3.10 with $SNR = 10dB$

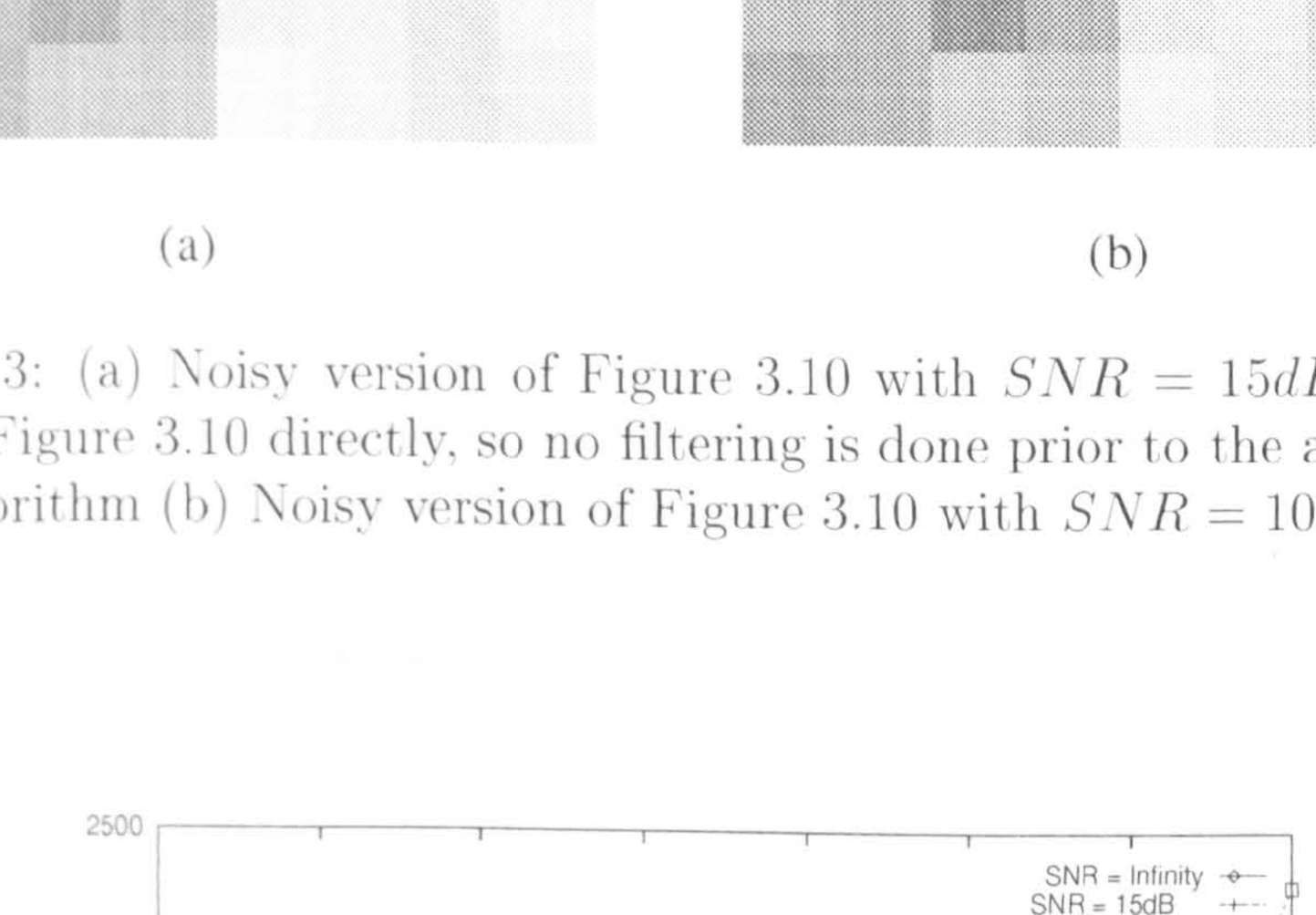


Figure 3.14: The relationship between C and total iterations when different amounts of noise are added to the clean image (I = 10)

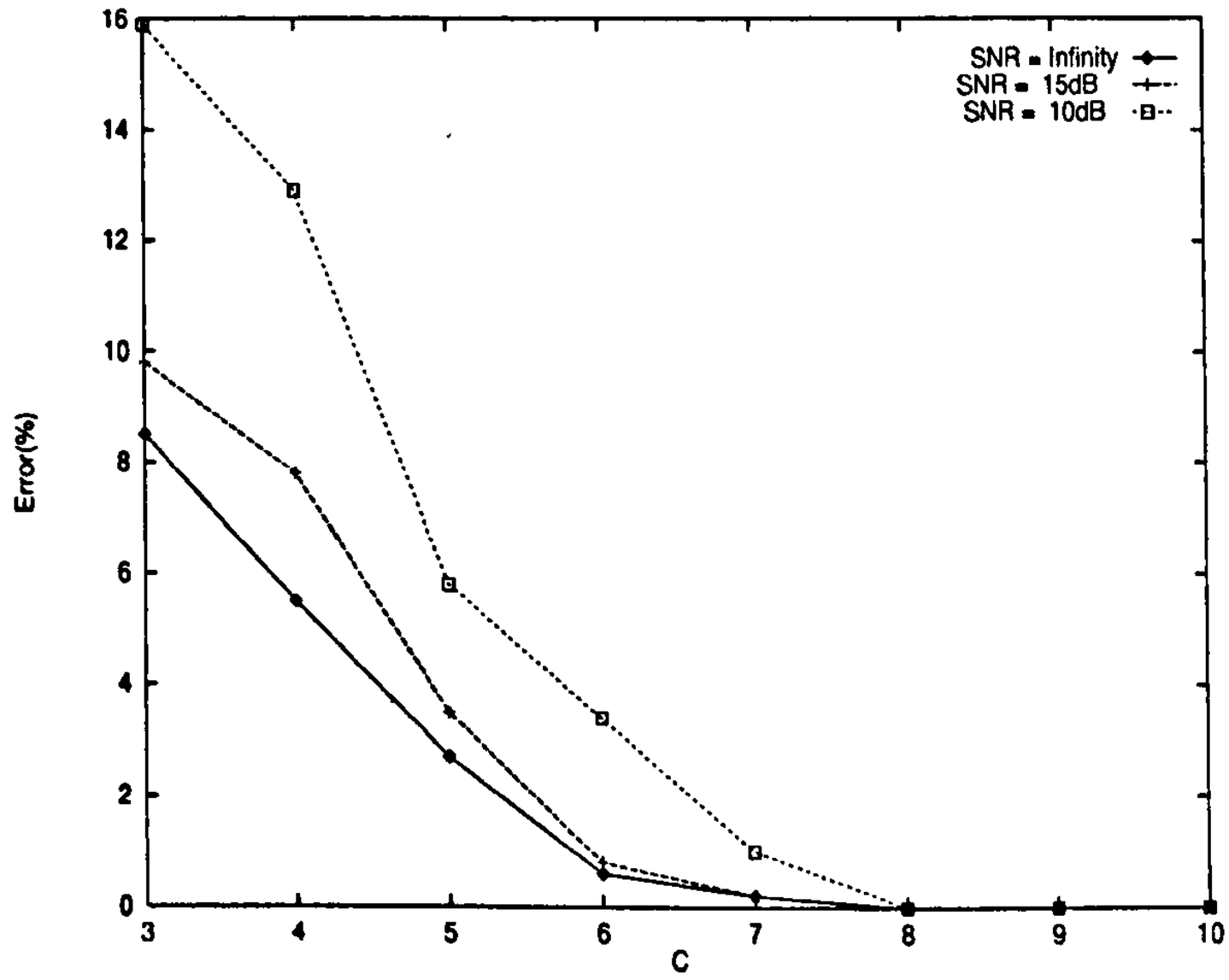


Figure 3.15: The relationship between C and error rate when different amount of noise are added to the clean image (I = 10)

3.6 Conclusions

The experiments carried out in the above examples show that MMRF formulation has the potential to yield good segmentation results at reasonable computational cost. Statistics about the computational cost in terms of the number of iterations per pixel will be given in Chapter 4 and 5 where the algorithm is performed without supervision. By adopting a multiresolution approach, class-position uncertainty is alleviated and the conditioning of lower levels on higher ones, together with the local characteristics of MRF's keep the computational cost down. It therefore seems an appropriate model to use for texture segmentation.

Chapter 4

Region-Based Texture Segmentation

In this chapter, we present an algorithm which partitions a textured image into regions, with each possessing homogeneous texture properties. The algorithm consists of a texture feature extraction stage, followed by a segmentation one. The Multiresolution Fourier Transform (MFT) is first applied to a high-pass version of a textured image to create an image pyramid as described in Section 3.4. A set of four spatially localised texture features is then extracted from each block at each level of the image pyramid to describe it. Each set of texture features is based on a two-component model of texture, in which one component is an affine deformation, representing the structural or deterministic element and the other is a stochastic one based on the local Fourier energy spectrum [49]. Based on the extracted feature sets, the image pyramid is modelled as a sequence of MRF's and stochastic relaxation is adopted to maximise the posterior probability in assigning class labels to the blocks (sites) being visited. Class information is propagated from low spatial resolution to high spatial resolution, via appropriate modifications to the interaction energies defining the field, to minimise class-position uncertainty. Experiments on the segmentation of natural textures are used to show the

potential of the method.

4.1 Feature Extraction — A Two-Component Model

The task of texture feature extraction involves three major steps:

1. High-pass filtering using the Laplacian pyramid [13].

2. Multiresolution Fourier Transform (MFT) of the filtered image pyramid.

3. Extracting features from the MFT.

4.1.1 Image Filtering

In order to help detecting the texture boundaries, we want the boundaries — the 'discontinuity' — to be emphasised. Also, since textures consist of primitives and tiny edges (intensity fluctuation) composing the primitives, we want to have this important information being enhanced as well. The texture boundaries and the tiny edges within textures are all high frequency components of the image. The purpose of high-pass filtering an image is to reduce the influence of low frequency components of the image while preserving and enhancing the high frequency information. We utilise a method similar to Burt et al's [13] Laplacian Pyramid to get a high-pass filtered version of the original image. We first build a multiresolution gray level Gaussian Pyramid based on the original image (the bottom level) with each level of the hierarchy created by convolving a Gaussian weighting kernel [131] with the level *below*. The weighting kernel employed in the thesis is shown in Table 4.1. The size of the kernel used in Burt et al's work [13] is 5×5 while the

Table 4.1: A Gaussian kernel for building a gray level Gaussian Pyramid

0.0109	0.0582	0.0582	0.0109
0.0582	0.1227	0.1227	0.0582
0.0582	0.1227	0.1227	0.0582
0.0109	0.0582	0.0582	0.0109

size of the kernel used in this work is 4×4. We tailored the size to make it consistent with Multiresolution Fourier Transform where the size of the sampling window is $2^k \times 2^k$. The values of the elements of the kernel can vary provided Equation (4.1) is observed

$$1 = \sum_{m=0}^{3} \sum_{n=0}^{3} w(m,n) \tag{4.1}$$

where $w(m,n)$ is the (m,n)th element of the kernel, to give a gain at d.c. of unity.

Thus given $g_N(i,j)$ as the original image, a sub-sampling process can be applied to create a Gaussian Pyramid as shown in Figure 4.1 with $g_N(i,j)$, the original image, as the bottom level and $g_k(i,j)$ on top of $g_{k+1}(i,j)$.

$$g_k(i,j) = \sum_{m=0}^{3} \sum_{n=0}^{3} w(m,n)\, g_{k+1}(2i+m, 2j+n), \quad 0 \le k < N \tag{4.2}$$

Although the dimension of the kernel is 4×4, it slides in 2-pixel wide steps in both horizontal and vertical direction, i.e., the sampling density is decreased by a factor of 4, so the size of the 2-D image $g_k(i,j)$ is a quarter of $g_{k+1}(i,j)$. That is to say that the areas covered by two consecutive operation of the kernel are *half-overlapped*. This is to serve the purpose of reducing aliasing artifacts.

The result of this local averaging operation on each pixel at each level is a low-pass filtered version of the level below. So we can expect that the higher

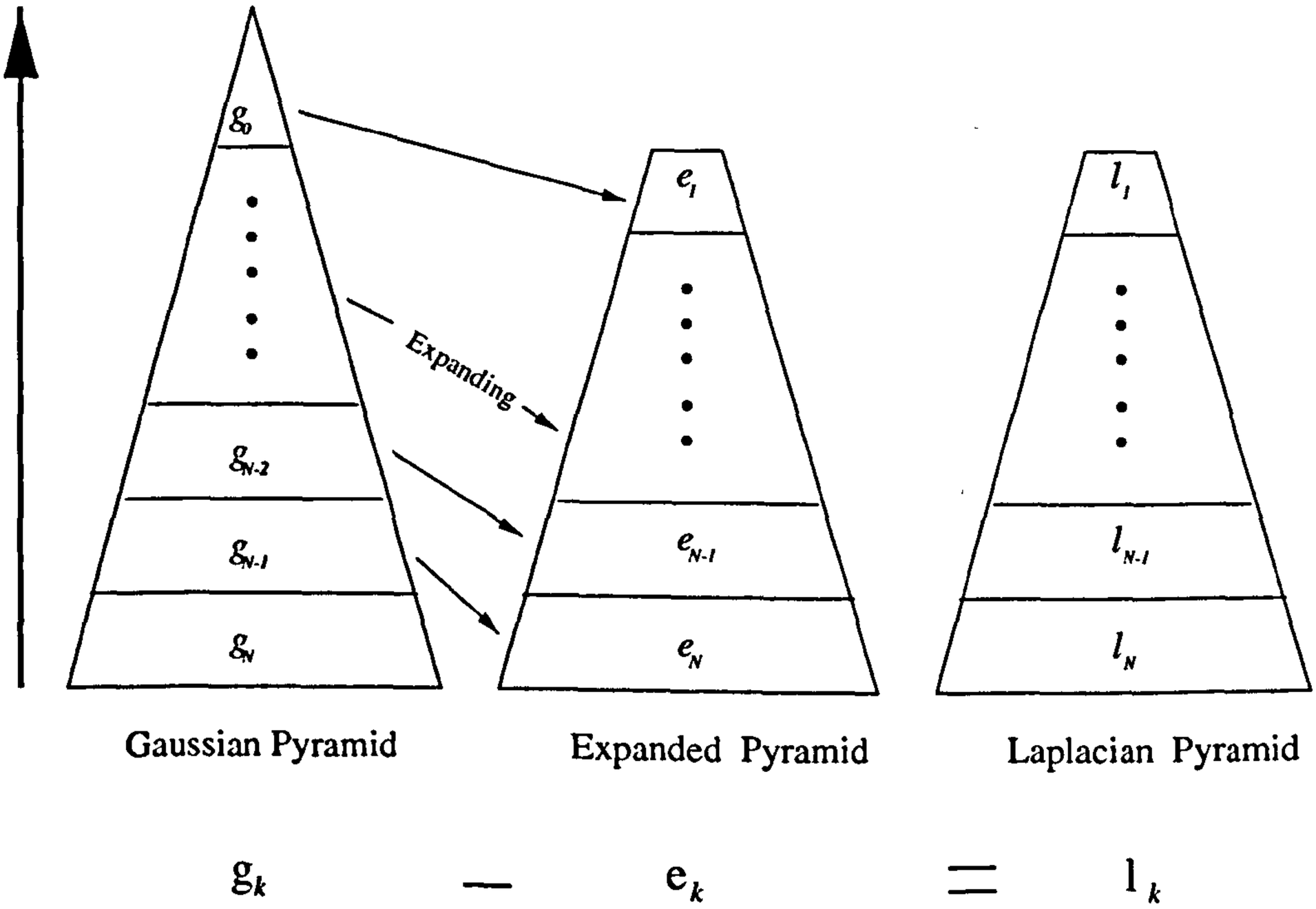


Figure 4.1: From Gaussian Pyramid to Laplacian Pyramid

the level in the Gaussian Pyramid, the lower the variance of the features within the same texture regions, i.e., the higher resolution in class space. However, the spatial resolution is reduced during the convolution process, i.e., the position of the textural boundaries becomes less certain.

Now if we expand any level of the Gaussian Pyramid to the size of the immediate level below using Equation (4.3), we end up with an Expanded Pyramid as shown in Figure 4.1.

$$e_k(x,y) = 4 \sum_{m=0}^{3} \sum_{n=0}^{3} w(m,n)\, g_{k-1}(\frac{x+m}{2}, \frac{y+n}{2}), \quad 0 < k \leq N \qquad (4.3)$$

where e_k is the expanded version of g_{k-1} with size equal to that of g_k. Since only the four terms for which $\frac{x+m}{2}$ and $\frac{y+n}{2}$ are integers are included in Equation (4.3), the sum is multiplied by 4 to reflect the fact that there are 16 pixels covered by the weighting function. A high-pass image pyramid called

Laplacian Pyramid (see Figure 4.1) can now be created with each pixel at each level taking the difference between the gray levels at the corresponding pixels at each level of the Gaussian Pyramid and Expanded Pyramid [13], that is

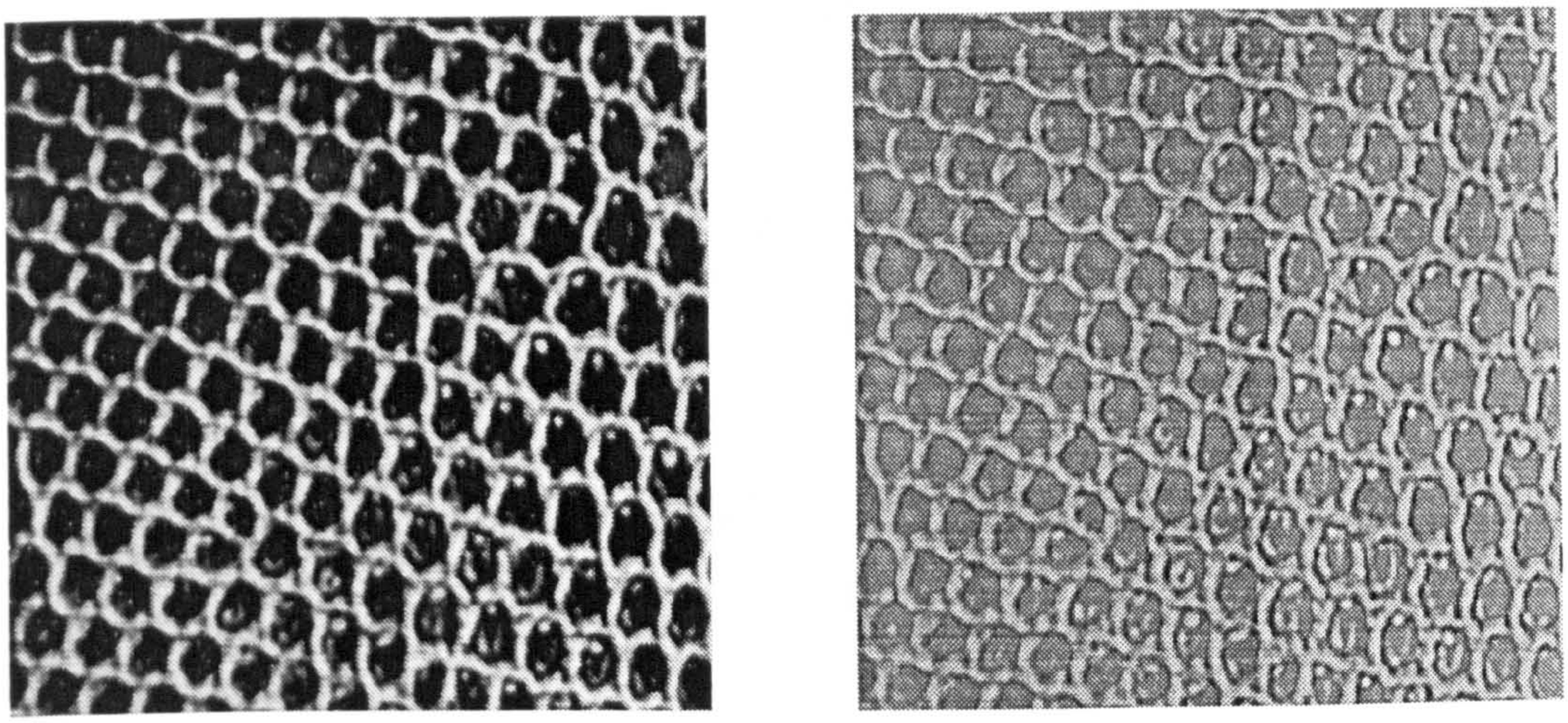
$$l_k(x,y) = g_k(x,y) - e_k(x,y) \tag{4.4}$$

where $l_k(x,y)$ is the gray level of pixel (x,y) at level k of the Laplacian Pyramid.

It can be readily observed that the original image $g_N(x,y)$ can be recovered by expanding l_1 once and adding it to l_2, then expanding this image sum once and adding it to l_3, and so on until level N is reached. In this work, l_N is taken as the high-pass filtered version of the original image. If we want to preserve more information than l_N, we expand any level of the Gaussian Pyramid higher than level $N-1$ to the size of the original image and then take the difference between g_N and the newly expanded image as the high-pass filtered version. Figure 4.1 shows the process of establishing a Laplacian Pyramid. Figure 4.2(a) and (b) show a test image and its high-pass filtered version, respectively. Note that for the sake of perception, the gray levels of the high-pass image in Figure 4.2(b) have been raised by 128 so that all the pixels with gray level lower than 128 have a negative gray value in the real image while the pixels with gray level higher than 128 have positive gray value.

4.1.2 Multiresolution Fourier Transformation

Adopting the Multiresolution Fourier Transform (MFT) derived by Wilson et al. [132], we create an image pyramid with K resolution levels based on the original image of size $M \times M = 2^K \times 2^K$ as shown in Figure 4.3. Each



(a) Original image

(b) High-pass version of the original image

Figure 4.2: A test image and its high-pass filtered version

level k consists of an array of $2^k \times 2^k$ square blocks taken from the original image by sliding a sampling window of size $2^{K-k} \times 2^{K-k}$ over the image in 2^{K-k-1}-pixel wide steps in both horizontal and vertical directions. This means that the adjacent windows in either horizontal or vertical directions are half-overlapped. For example at level 0, the top level, the size of the sampling window is exactly the same as that of the original image, therefore the sampling results in only one block at that level. At the bottom level of the pyramid, the sampling window size is only one pixel, thus the sampling result is the duplicate of the original image. Since the size of the window used in a level is a quarter of the one used at the immediate ancestor level, the pyramid thus created conforms to a quad-tree structure. The 50 % overlap ensures that the measurements of interest estimated from each block vary smoothly across the image. This multiresolution structure provides a trade-off between *where* the boundaries are and *what* the classes are. The multiresolution Fourier transform is done by transforming the windowed

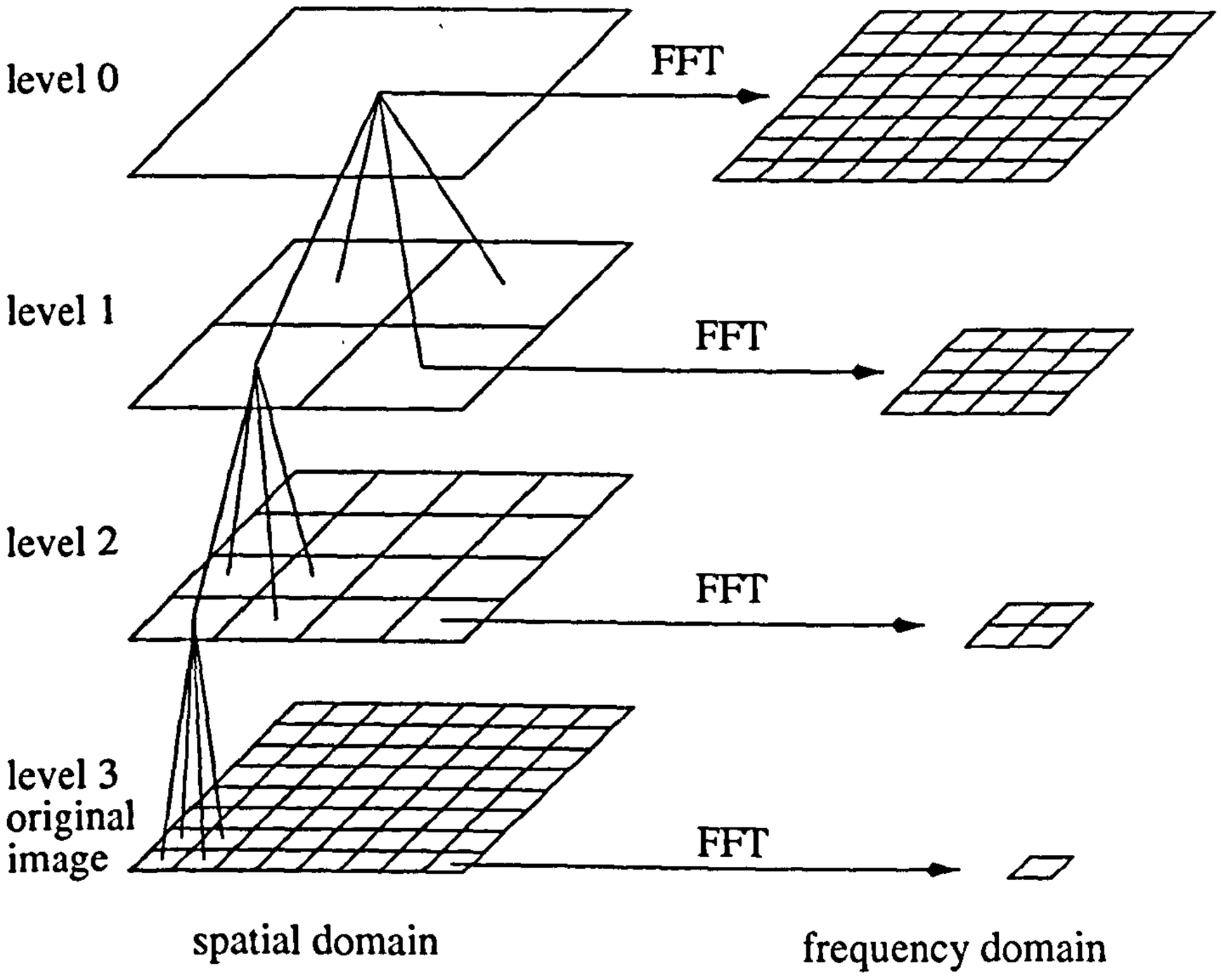


Figure 4.3: Structure of Multiresolution Fourier Transform (MFT)

blocks at each level into the frequency domain [132].

A MFT coefficient at level k is identified by three parameters: spatial co-ordinate $\vec{\xi}(k)$, frequency co-ordinate $\vec{\omega}(k)$ and scale $\gamma(k)$. A 2-D MFT coefficient can be represented by

$$\hat{f}(\vec{\xi}(k),\vec{\omega}(k),\gamma(k)) = \sum_{\vec{\xi}^{\prime}(k)} w_n(\vec{\xi}(k)-\vec{\xi}^{\prime}(k),\gamma(k))\ f(\vec{\xi}^{\prime}(k))\ e^{[-j\vec{\xi}^{\prime}(k)\cdot\vec{\omega}(k)]} \quad (4.5)$$

where $f(\vec{\xi}^{\prime}(k))$ represents the original $M \times M$ image in the spatial domain and $w_n(\vec{\xi}(k)-\vec{\xi}^{\prime}(k))$ is a windowing function with maximal energy concentration in both spatial and spatial frequency domain. Therefore, each level of MFT resembles a 2-D Short Time Fourier Transform (STFT). In order to reduce artifacts, in this work we employ a sine function as the windowing function,

i.e., at level k we denote

$$\begin{aligned} w_n(\vec{\xi}(k)) &= w_n(x, y, k) \\ &= \sin\left(\frac{\pi}{2^{K-k}}\, x\right) \sin\left(\frac{\pi}{2^{K-k}}\, y\right) \quad , 0 \leq x, y < 2^{K-k} \end{aligned} \tag{4.6}$$

where (x, y) is the co-ordinates within a block. This makes inversion of a given level of the MFT simple and has been shown to be adequate in terms of spectral leakage [15].

The Multiresolution Fourier Transform has the following properties which are important in image analysis [15][49]:

- Linearity: MFT is a hierarchical structure of STFT's, and because STFT is linear, the MFT inherits the linear property.

- Locality: each level of MFT performs local operation in each domain. This locality property enables the segmentation algorithm to minimise uncertainty.

- Invertibility: Since STFT is invertible, the MFT has this property. This prevents errors due to information loss during the transformation from one domain to another.

Note that for an image of size $M \times M$, because we need to have enough samples of blocks to do the segmentation, the building process of a MFT structure has to stop at a nominal top level k with $2^k \times 2^k$ blocks, each covering an area of $2^{K-k} \times 2^{K-k}$ pixels. Also since texture is a regional property, a small block is not big enough to contain reliable textural features, so that the block size of the nominal bottom level of the MFT structure should be larger than 1×1. In our experience a nominal top level should at least contain 8×8

sites while a level with blocks covering 8×8 pixels is a reasonable choice for the nominal bottom level.

4.1.3 Extracting Features

As noted in Chapter 2, most attempts to segment or classify textures are based on either statistical or structural descriptions [45]. Statistical approaches such as co-occurrence matrices and auto-regressive models represent texture with statistics extracted from local image measurements. Generally, they are good for textures with random spatial arrangements, but are not capable of characterising the structural information. On the other hand, structural approaches are utilised to analyse deterministic texture consisting of similar primitives spatially arranged according to some set of well-defined placement rules. However, in reality, natural textures seldom consist of identical primitives grouped by rigid placement rules, so structural approaches are not frequently employed. Segmentation algorithms based on the texture features extracted using either of these approaches alone frequently lead to under-segmentation, due to the inadequacy of discriminatory information about the texture classes. It is clearly preferable to combine both statistical and structural feature extraction techniques to estimate stochastic and deterministic features. To this end, we present in this section a novel approach which extracts four local measurements to describe texture features based on the 'stochastic+deterministic' decomposition, which is a generalisation of the Wold decomposition of signals [49]. They are:

$X_{ss'1}$: Squared difference between the average gray levels of two neighbour blocks.

$X_{ss'2}$: Difference in the spectral energy densities, the stochastic component,

of two neighbouring sites via Multiresolution Fourier Transform,

$$X_{ss'2} = \sum_{\vec{\omega}} (|\hat{f}_s(\vec{\omega})| - |\hat{f}_{s'}(\vec{\omega})|)^2$$

where $\hat{f}_.(\vec{\omega})$ is the DFT of $f_.(\vec{\xi})$ and $\vec{\omega}$ is the coordinate in frequency domain.

$X_{ss'3}$, $X_{ss'4}$: Two measures associated with the deterministic component, based on an affine deformation model [49]

$$f_s(\vec{\xi}) = f_{s'}(A^{-1}(\vec{\xi} - \vec{\chi})) + \nu_s(\vec{\xi}) \tag{4.7}$$

where site s' is a 4-neighbour of site s, $f_s(\vec{\xi})$ is a sub-image at site s, A is that 2×2 nonsingular linear co-ordinate transform and $\vec{\chi}$ that translation which together give the best fit in terms of total deformation energy between the two patches.. $\vec{\xi}$ is the coordinates of pixels within the sub-image at a site in the spatial domain. A is identified using the method described in [49].

The two measures associated with the deterministic component can now be defined as:

$X_{ss'3}$: The deformation term $\|A - I\|^2$ represents the amount of 'warping' required to match the given patch using its neighbour. I is the identity matrix. If $\|A - I\|^2 = 0$, there is no 'warping' error after the affine transformation.

$X_{ss'4}$: The error term $\|\nu_s(\vec{\xi})\|^2$ is the average residual error in the approximation, i.e.,

$$\begin{aligned} X_{ss'4} &= \|\nu_s(\vec{\xi})\|^2 \\ &= \sum_{\vec{\xi}} |f_s(\vec{\xi}) - f_{s'}(A^{-1}(\vec{\xi} - \vec{\chi}))|^2 \end{aligned}$$

To identify $\boldsymbol{A}$, a pair of centroid vectors is first calculated to represent the local Fourier spectrum of each block, at the appropriate scale using the MFT. To find the centroid vector pair, $(\vec{\mu}_1, \vec{\mu}_2)$, of a sub-image site $f_s(\vec{\xi})$, the values of two variable angles θ_1 and θ_2 are to be determined such that

$$\sigma^2(\theta_1,\theta_2) = \frac{M_1(\theta_1,\theta_2)\,\sigma_1^2(\theta_1,\theta_2) + M_2(\theta_1,\theta_2)\,\sigma_2^2(\theta_1,\theta_2)}{M_1(\theta_1,\theta_2) \;+\; M_2(\theta_1,\theta_2)} \tag{4.8}$$

is minimum, where

$$M_i(\theta_1,\theta_2) = \sum_{\vec{\omega}\in\Theta_i(\theta_1,\theta_2)} |\hat{f}(\vec{\omega})|^2 \qquad ,i \;=\; 1,2 \tag{4.9}$$

$$\sigma_i^2(\theta_1,\theta_2) = \frac{1}{M_i(\theta_1,\theta_2)} \sum_{\vec{\omega}\in\Theta_i(\theta_1,\theta_2)} |\hat{f}(\vec{\omega})|^2 \,\|\vec{\omega} - \vec{\mu}_i(\theta_1,\theta_2)\|^2, i = 1,\, 2 \tag{4.10}$$

$\mu_i(\theta_1,\theta_2)$ is the centroid vector

$$\mu_i(\theta_1,\theta_2) \;=\; \frac{1}{M_i(\theta_1,\theta_2)} \sum_{\vec{\omega}\in\Theta_i(\theta_1,\theta_2)} |\hat{f}(\vec{\omega})|^2\,\vec{\omega}\;, i \;=\; 1,\, 2 \tag{4.11}$$

and $\Theta_1(\theta_1,\theta_2)$ and $\Theta_2(\theta_1,\theta_2)$ are the sets of coordinates in each of the two segments of the half-plane starting at angle θ_1 and divided at angle θ_2 (see Figure 4.4). To find the minimum $\sigma^2(\theta_1,\theta_2)$, each combination of angles θ_1 and θ_2 is tested. The intervals of θ_1 and θ_2 are $0 \leq \theta_1 < \pi$ and $\theta_1 < \theta_2 < \theta_1 + \pi$. Note that by taking advantage of the Hermitian symmetry of the Fourier transform, only a half plane of the local spectrum/region needs to be analysed. In Figure 4.4, θ_1 is used to divide the local spectrum into two half-planes and θ_2 is used to sub-divide the half-plane starting at θ_1 into two segments, Θ_1 and Θ_2.

Now, for any two sub-image blocks of interest, f_s and $f_{s'}$, with their representative centroid pairs $(\vec{u}_1, \vec{u}_2)$ and $(\vec{v}_1, \vec{v}_2)$, respectively, the co-ordinate transform $\boldsymbol{A}$ is calculated from

$$\vec{u}_i = \boldsymbol{A}\vec{v}_i \qquad\qquad i = 1,2 \tag{4.12}$$

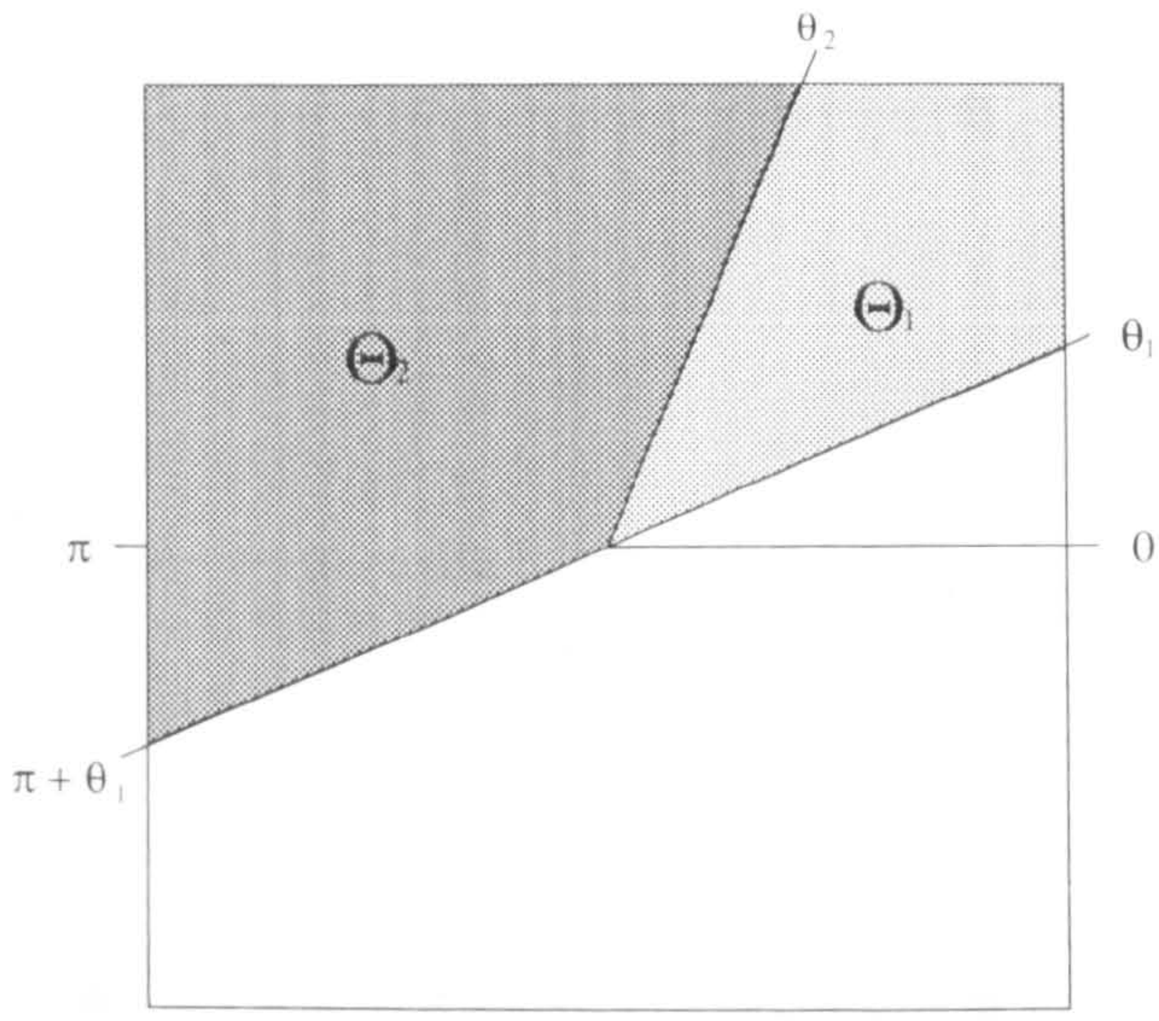


Figure 4.4: Division of a spectral half-plane into two segments by θ_1 and θ_2.

Once $\boldsymbol{A}$ is calculated, $f_{s'}$ is affine transformed by transforming all of its co-ordinates into a new set of co-ordinates. The new co-ordinates are not necessarily discrete-valued. Bilinear interpolation is utilised to calculate the intensity at each discrete co-ordinate using the four nearest neighbouring real-valued co-ordinates (see Figure 4.5). This method is given as

$$\begin{aligned} f(x', y') &= (1-a)(1-b)f(x,y) + a(1-b)f(x+1,y) \\ &\quad +(1-a)bf(x,y+1) + abf(x+1,y+1) \end{aligned} \tag{4.13}$$

where $f(.)$ is the intensity, (x', y') is a discrete-valued co-ordinate, (x, y) is a real-valued co-ordinate, and

$$a = x' - x$$
$$b = y' - y$$

Calway [15] has shown that Bilinear interpolation is adequate for estimating the (complex) values at arbitrary discrete-valued frequency co-ordinates based on the MFT sampling scheme.

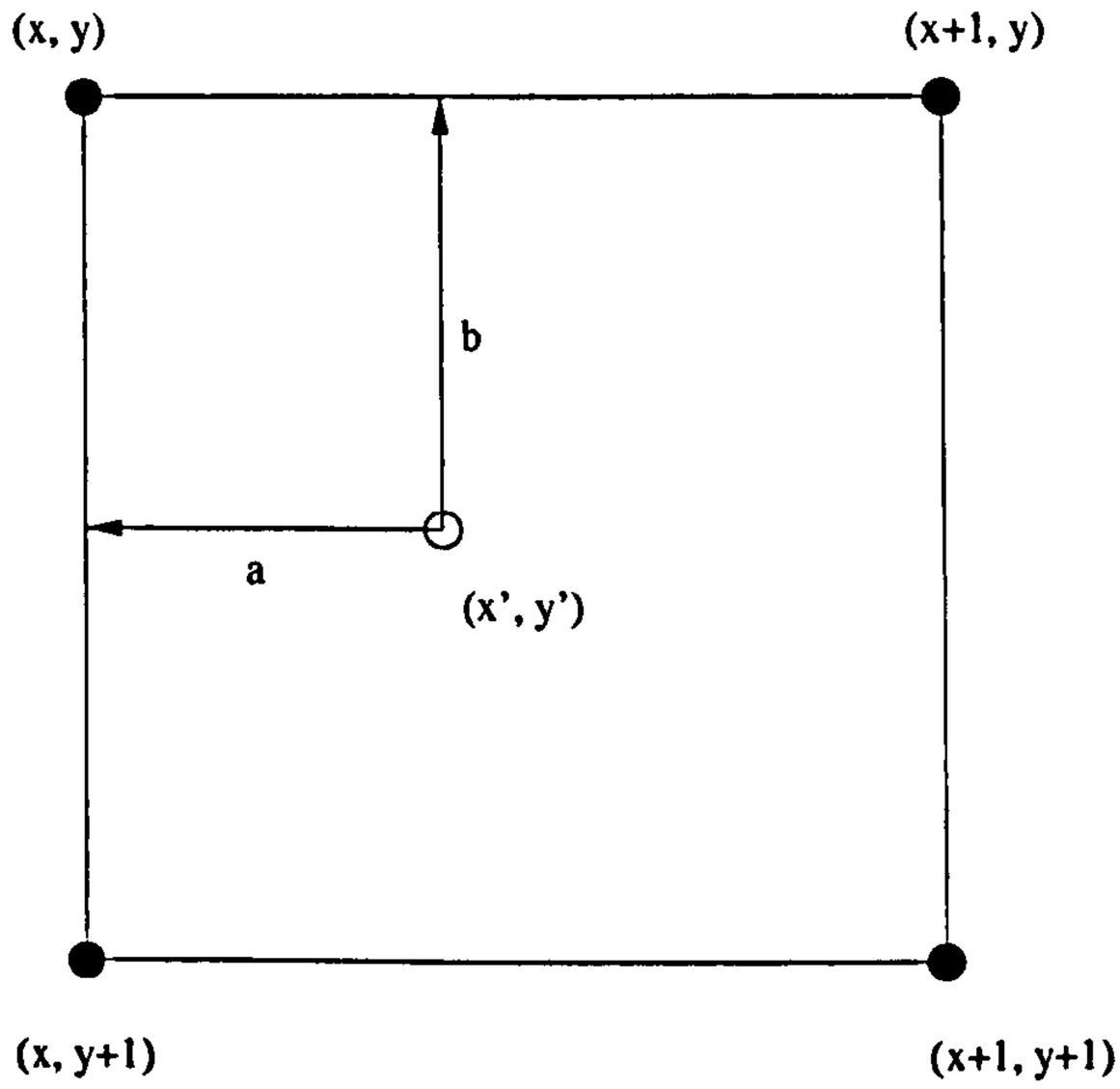


Figure 4.5: Interpolating the value at (x', y') from the value of its four nearest neighbouring real-valued co-ordinates

4.2 Segmentation Using MMRF's

We model the MFT structure as a sequence of MRF's, whose underlying theory has been introduced in Chapter 3. The neighbourhood of the MRF model we impose in the region-based process consists of the 4 first-order neighbours on the same resolution level and the *father* on the level above as shown in Figure 3.5, i.e.,

$$\begin{aligned} \mathcal{N}_s &= \{(i-1,j,k),(i+1,j,k),(i,j-1,k), \\ &\qquad (i,j+1,k),(\lfloor i/2 \rfloor, \lfloor j/2 \rfloor, k-1)\} \end{aligned} \tag{4.14}$$

where (i,j) are the coordinates of site s and $\lfloor . \rfloor$ denotes the floor of a real number. The clique system within $\mathcal{N}_s$ is $\mathcal{C}_2$ only (see Equation (3.19)).

The segmentation algorithm using the MMRF framework starts at a nominal top level. Due to *ergodicity*, the property of the sampler that the eventual class label configuration of the image is independent of the initial configu-

ration, at the nominal top level, the class labels of the pixels in the image may be randomly assigned. Upon convergence of the algorithm, i.e., no more label changes after a full raster scan, the detected texture boundaries and class labels are propagated down to the next level so as to allow refinement to be carried out at the new level. Since the class resolution is higher at higher levels, to accelerate the computation, we simply take the *1 to 4* expansion of the final configuration of the immediate ancestor level of a lower level as its initial configuration. In turn, since the initial configuration at lower levels is a coarse segmentation instead of a random one, the algorithm is allowed to start from a temperature lower than the starting temperature of the immediate ancestor level. However, there is no criterion about how much lower the starting temperature can be set at a new level.

4.2.1 Definition of Interaction Energy

The interaction energy in Equation (3.16) between a site and its neighbourhood defining the MRF is a function of the four local measurements describing texture features and is defined as

$$U(\lambda_s, \lambda_{\mathcal{N}_s}, X_{\mathcal{N}_s m}) = \sum_{m=1}^{4} \alpha_m \sum_{s' \in \mathcal{N}_s} V_m(\lambda_s, \lambda_{s'}, X_{ss'm}) \tag{4.15}$$

where $\lambda_s, 1 \leq \lambda_s \leq L$ is the class label at site s, $X_{\mathcal{N}_s}$ represents the measurements over the sites within the neighbourhood $\mathcal{N}_s$ and the sum over m allows for 4 texture descriptors to be included. The pairwise interaction potential V_m is a suitably defined function of class labels of the two neighbouring sites s and s' and the measurement $X_{ss'm}$ estimated from the two, α_m is the weight of V_m. Note that, at the nominal top level, no interaction between site s and its father is involved in the calculation of V_m and for the other levels, only pairwise interaction energy V_1 between site s and its father

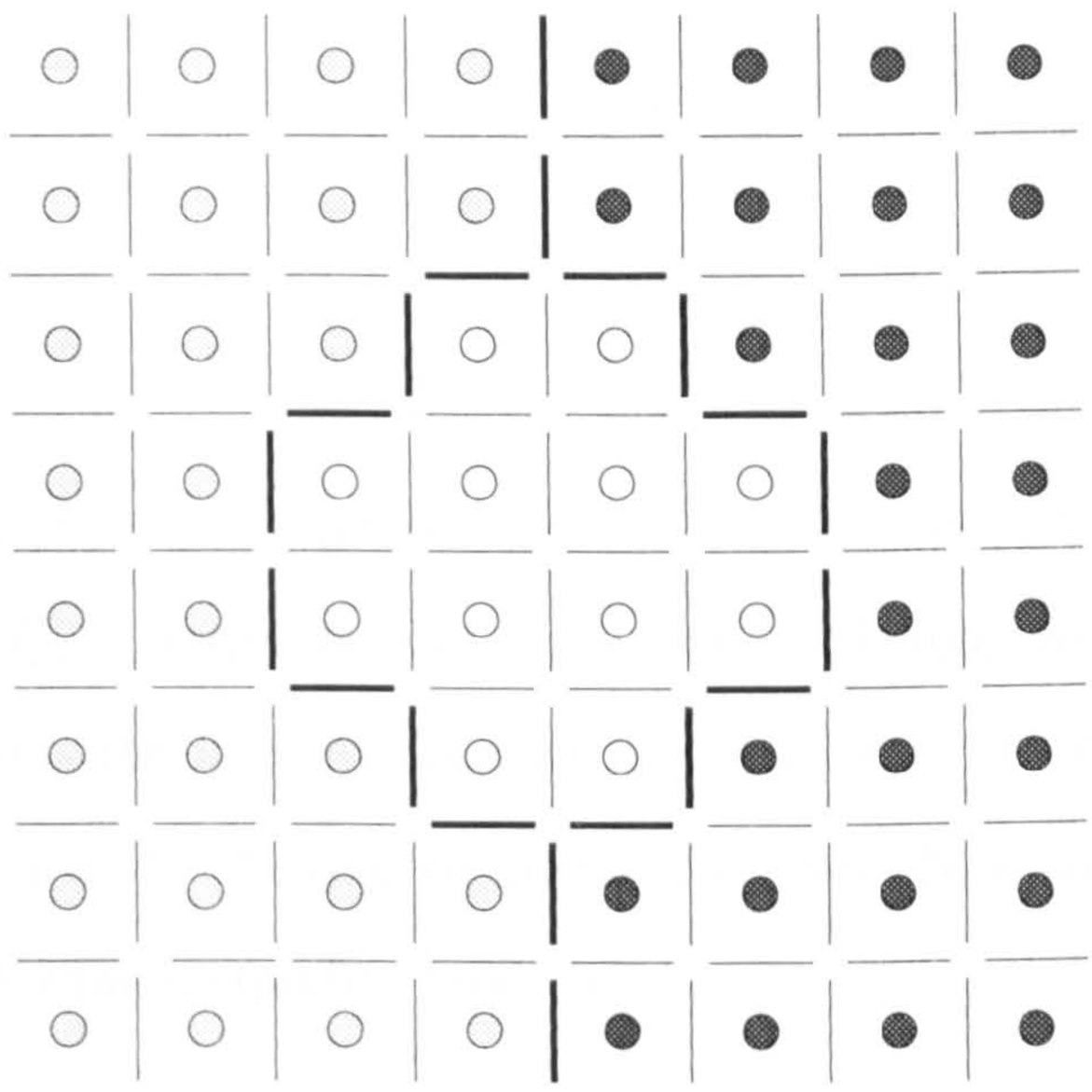
is involved in the calculation of $U(\lambda_s, \lambda_{\mathcal{N}_s}, X_{\mathcal{N}_s m})$ because $X_{ss'1}$ is the only measurement estimated between the two. Before we give the definition of $V_m(\lambda_s, \lambda_{s'}, X_{ss'm})$, let us first look at the probability distribution of $X_{ss'm}$. Using the symbols defined in Section 3.2, if $X_{ss'm}$ represents the measure mentioned in Section 4.1.3, we denote

$$\sigma_{wm}^2 = \sum_{\lambda_s = \lambda_{s'}} X_{ss'm} \quad , \forall s \text{ and } s' \tag{4.16}$$

$$\sigma_{om}^2 = \sum_{\lambda_s \neq \lambda_{s'}} X_{ss'm} \quad , \forall s \text{ and } s' \tag{4.17}$$

Figure 4.6 shows an image with three different regions. Each circle in the figure represents a site. The *thick* bars between neighbouring sites stand for strong feature measurements $X_{ss'm}$ between the site pairs belonging to **different** classes while the *thin* bars stand for small $X_{ss'm}$ between sites belonging to the **same** region. From this figure, we know that σ_{wm}^2 is simply the average value of the thin bars while σ_{wo}^2 is simply the average value of the thick ones. They are both re-calculated at the beginning of each iteration because of their dependence on the varying configuration.

To examine the distribution of the measurements, we take 1984 pairs of blocks of 16×16 pixels from Figure 1.1(a) and (d), respectively, and calculate the differences between the mean gray levels of the two blocks of each pair. We call these differences *intra-region differences.* We then take 1984 pairs of blocks, each pair consisting of one block taken from Figure 1.1(a) and the other from Figure 1.1(d), and calculate the differences between the mean gray level of the two blocks of each pair. We call these differences *inter-region differences.* The probability distribution functions (histograms) of the two kinds of differences are displayed in Figure 4.7. The shapes of the two histograms conform roughly to Gaussian distributions with different means



——— : Small difference between sites

——— : Large difference between sites

Figure 4.6: Feature measurements between sites.

and variances. However, by and large, it is an ideal assumption that textured images possess two probability distribution functions of feature difference significantly differing in mean values. In reality, textured images typically have both probability distribution functions centred at zero mean as shown in Figure 4.8 — the worst case. To ensure that the algorithm can handle the worst case, we assume that the two measurements have zero mean and denote

$$P(X_{ss'm}|\lambda_s = \lambda_{s'}) = \frac{1}{\sigma_{wm}\sqrt{2\pi}}\, e^{-\frac{X_{ss'm}}{2\sigma_{wm}^2}} \tag{4.18}$$

$$P(X_{ss'm}|\lambda_s \neq \lambda_{s'}) = \frac{1}{\sigma_{om}\sqrt{2\pi}}\, e^{-\frac{X_{ss'm}}{2\sigma_{om}^2}} \tag{4.19}$$

Equation (4.18) is the joint probability distribution of the random variable difference $X_{ss'm}$ assuming both sites s and s' belong to the *same* class. Equation (4.19) is the joint probability distribution of the random variable $X_{ss'm}$ assuming s and s' belong to *different* classes. Now we want to know whether the following inequality holds or not:

$$P(\lambda_s = \lambda_{s'}|X_{ss'm}) > P(\lambda_s \neq \lambda_{s'}|X_{ss'm}) \tag{4.20}$$

If Inequality (4.20) holds, it is more likely that sites s and s' belong to the same class. Otherwise they belong to different classes with higher probability. According to Bayes's theorem, (4.20) is equivalent to

$$\frac{P(X_{ss'm}|\lambda_s = \lambda_{s'})\, P(\lambda_s = \lambda_{s'})}{P(X_{ss'm})} > \frac{P(X_{ss'm}|\lambda_s \neq \lambda_{s'})\, P(\lambda_s \neq \lambda_{s'})}{P(X_{ss'm})} \tag{4.21}$$

Substituting Equations (4.18) and (4.19) into (4.21), we obtain

$$\ln\left(\frac{P(\lambda_s = \lambda_{s'})}{\sigma_{wm}}\right) - \frac{X_{ss'm}}{2\sigma_{wm}^2} > \ln\left(\frac{P(\lambda_s \neq \lambda_{s'})}{\sigma_{om}}\right) - \frac{X_{ss'm}}{2\sigma_{om}^2} \tag{4.22}$$

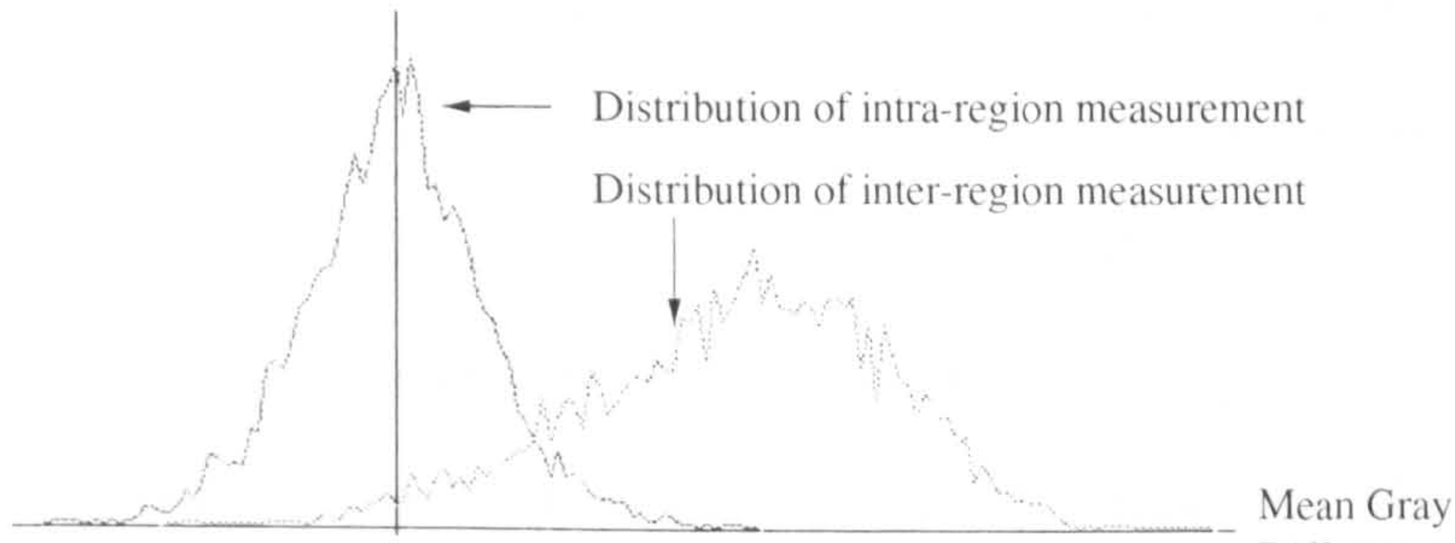


Figure 4.7: Distributions of intra- and inter-region feature measurements.

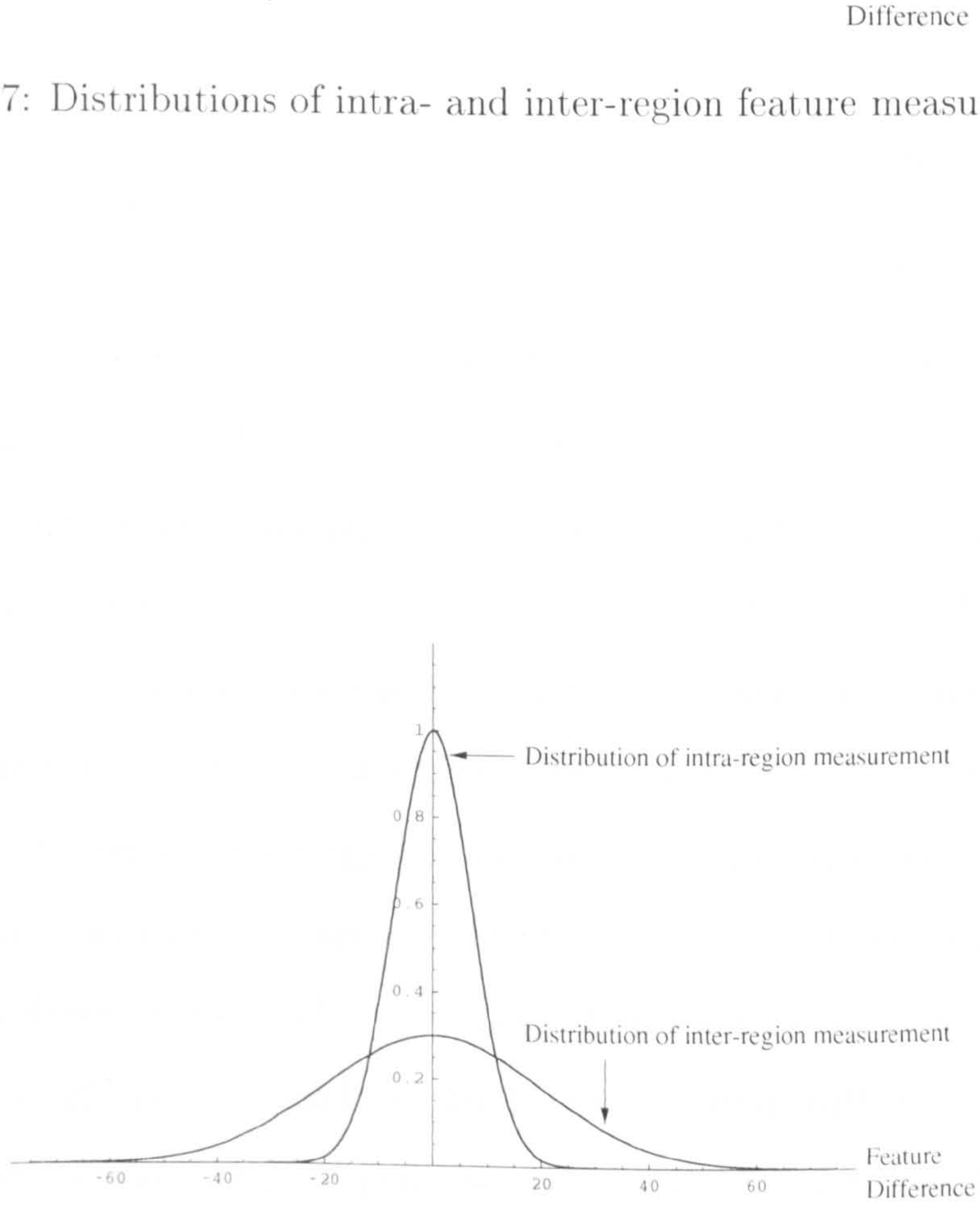


Figure 4.8: Distributions of intra- and inter-region feature measurements.

As we mentioned previously that given the class labels of the neighbouring sites of s, we want to encourage the assignment of a *correct* label with a low interaction energy (Equation (4.15)) to site s, employing either side of Inequality (4.22), depending on the value of λ_s and $\lambda_{s'}$, as the potential $V_m(\lambda_s, \lambda_{s'}, X_{ss'm})$ is a reasonable definition. If $\lambda_s = \lambda_{s'}$, the potential is equal to the *right-hand* side of Inequality (4.22), otherwise, the *left-hand* side.

$$V_m(\lambda_s, \lambda_{s'}, X_{ss'm}) = \begin{cases} \ln\left(\frac{P(\lambda_s \neq \lambda_{s'})}{\sigma_{om}}\right) - \frac{X_{ss'm}}{2\sigma_{om}^2} & , if\ \lambda_s = \lambda_{s'} \\ \ln\left(\frac{P(\lambda_s = \lambda_{s'})}{\sigma_{wm}}\right) - \frac{X_{ss'm}}{2\sigma_{wm}^2} & , if\ \lambda_s \neq \lambda_{s'} \end{cases} \tag{4.23}$$

The physical meaning of Equation (4.23) is that if sites s and s' *do* belong to the same class, i.e., the probability that Inequality (4.22) holds is higher, it is more likely that assigning s and s' the same label will be encouraged by given a lower interaction energy. By contrast, if sites s and s' do *not* actually belong to the same class, i.e., the left-hand side of Inequality (4.22) is more likely to be smaller than the right-hand side, it is highly possible that a larger interaction energy will be given to discourage the assignment of the same label to sites s and s'. On the other hand, if sites s and s' are assigned different class labels, while they *do* belong to the same class, i.e. Inequality (4.22) is more likely to hold, the algorithm will tend to impose a larger interaction energy to penalise the assignment of different labels to sites s and s'. By contrast, if they do not belong to the same class, the algorithm will tend to endorse the assignment of different labels to the sites with a lower interaction energy. A similar approach is adopted for the other 3 measurements, a computationally convenient approximation which is not unmeasurable, given their forms.

The derivation of potential above is only valid for site s and its 4-neighbours at the same resolution level. To derive the potential between site s and its father site f, some parameters have to be elaborated. As we mentioned previously the only feature measurement between site s and its father f employed in the algorithm is the squared difference, X_{sf}, between the gray level, X_s, of site s and the mean gray level, X_f, of the region to which s's father belongs. Based on the same argument, the potential between any site s and its father f can be denoted as

$$V_f(\lambda_s, \lambda_f, X_{sf}) = \begin{cases} \ln\left(\frac{P(\lambda_s \neq \lambda_f)}{\sigma_{of}}\right) - \frac{X_{sf}}{2\sigma_{of}^2} & if\ \lambda_s = \lambda_f \\ \ln\left(\frac{P(\lambda_s = \lambda_f)}{\sigma_{wf}}\right) - \frac{X_{sf}}{2\sigma_{wf}^2} & if\ \lambda_s \neq \lambda_f \end{cases} \tag{4.24}$$

where $P(\lambda_s \neq \lambda_f)$ and $P(\lambda_s = \lambda_f)$ are the probabilities that $\lambda_s \neq \lambda_f$ and $\lambda_s = \lambda_f$, respectively, They are to be elaborated in Section 4.2.4.

$$X_{sf} = (X_s - X_f)^2 \tag{4.25}$$

where X_f is the mean gray level, calculated when the algorithm converges at level $k-1$, of the region to which s's father f belongs and

$$\sigma_{wf}^2 = \sum_{\lambda_s = \lambda_f} X_{sf} \quad , \forall s\ and\ f \tag{4.26}$$

$$\sigma_{of}^2 = \sum_{\lambda_s \neq \lambda_f} X_{sf} \quad , \forall s\ and\ f \tag{4.27}$$

However, the ranges of the four measurements $X_{ss'm}$ can be significantly different. In order to decide the weighting factor α_m in Equation (4.15) we have to scale all of the four different ranges to the same range ($[0, 255]$ in this work). The range scaling also serves the purpose of helping to decide the constant C in Equation (3.17) and partly solving the first issue addressed in Section 3.3.

Having scaled the range of measurement values, there is still no criterion for deciding the values of the weighting factor in Equation 4.15 for each measure. Generally, but not always, a higher ratio of $\sigma_{om}^2/\sigma_{wm}^2$ suggests a higher weighting factor. In our work, denoting $\beta_m = \sigma_{om}^2/\sigma_{wm}^2$, α_m is then decided as

$$\alpha_m = \frac{\beta_m}{\sum_{m=1}^{4} \beta_m} \quad , m = 1, 2, 3, 4 \tag{4.28}$$

4.2.2 Labelling the Sites

Initialisation Prior to the beginning of the algorithm at each level some variables are initialised as following:

- At the nominal top level:
  - λ_s is assigned randomly for any site s (ergodicity of the sampler)
  - σ_{wm}^2 = the average of the first three quartiles of the ascendantly ordered $X_{ss'm}$
  - σ_{om}^2 = the average of the last quartile of the ascendantly ordered $X_{ss'm}$
  - $P(\lambda_s = \lambda_{s'}) = 0.75$
  - $P(\lambda_s \neq \lambda_{s'}) = 0.25$

  The reason why $P(\lambda_s = \lambda_{s'})$ and $P(\lambda_s \neq \lambda_{s'})$ are thus initialised is because in general there are more intra-region sites than inter-region ones. Roughly speaking, each site is expected to have an edge on at most one side.

- At the levels other than the nominal top level:

- each site s is assigned the same label as its father
- For all 4-neighbouring sites, σ^2_{wm} and σ^2_{om} are calculated according to the initial configuration.
- σ^2_{wf} = the average of the X_{sf} between the sites which are not next to any side of the detected boundary and their father sites.
- σ^2_{of} = the average of the X_{sf} between the sites along both sides of the detected boundary and their father sites.
- $P(\lambda_s = \lambda_{s'})$, $P(\lambda_s \neq \lambda_{s'})$, $P(\lambda_s = \lambda_f)$ and $P(\lambda_s \neq \lambda_f)$ are calculated as to be described in Section 4.2.4.

After each iteration, $P(\lambda_s = \lambda'_s)$ and $P(\lambda_s \neq \lambda'_s)$ at the top nominal level, and σ^2_{wm} and σ^2_{om} at any level are updated according to the new configuration under the constraints that $P(\lambda_s = \lambda'_s) \geq P(\lambda_s \neq \lambda'_s)$ and $\sigma^2_{wm} < \sigma^2_{om}$. If the constraints are violated, the initial values are used instead.

Label Assignment The sites are visited in a raster scan order from top left to bottom right. To allow the algorithm to execute without supervision, we allow as many labels as the number of sites in each level to be used. However, calculating the probabilities of Equation (3.14) for all allowed labels at each site is prohibitively expensive in terms of computational cost. Actually, for all λ_s not in $\lambda_{\mathcal{N}_s}$, $P(\lambda_s|\lambda_{\mathcal{N}_s})$'s are all equal. Therefore, there is no need to calculate $P(\lambda_s|\lambda_{\mathcal{N}_s})$ for more than one λ_s which is not in $\lambda_{\mathcal{N}_s}$. Thus, for any site s, we only calculate the probabilities of the labels in $\lambda_{\mathcal{N}_s}$ and one which is not in $\lambda_{\mathcal{N}_s}$.

The advantage of this idea is twofold: first, it saves computational cost and solves the second issue mentioned in Section 3.3, by minimising the

number of labels to be involved. For example, for the worst case, assuming all the five neighbours (including father site) have different labels from each other, there are only six probabilities to be calculated. Knowing the maximal number of labels to be involved, in turn, helps us to decide an appropriate value of constant C in Equation (3.17), i.e. the first issue mentioned in Section 3.3 is partly solved. Secondly, this idea makes the specification or estimation of the number of textures unnecessary and enables the algorithm to work without supervision.

Suppose the set of the candidate labels to be assigned to any site s is

$$\begin{aligned} \lambda_N &= \lambda_{\mathcal{N}_s} \cup \{\text{any one } \notin \lambda_{\mathcal{N}_s}\} \\ &= \{\gamma_i | i = 1, 2, \ldots, n\} \end{aligned} \tag{4.29}$$

where n is the number of the elements in λ_N. Then after $P(\lambda_s = \gamma_i | \lambda_{\mathcal{N}_s})$, for all $i \leq n$, are calculated, we assign site s one of the labels from λ_N according to the output of an independent random number generator, r, taking value in the range $[0, 1]$. Suppose Figure 4.9 illustrates the values of $P(\lambda_s = \gamma_i | \lambda_{\mathcal{N}_s})$ for all i and let

$$P_i = \sum_{j=1}^{i} P(\lambda_s = \gamma_j | \lambda_{\mathcal{N}_s}) \tag{4.30}$$

then λ_s of site s is decided as

$$\lambda_s = \gamma_i \quad , \text{ if } P_{i-1} \leq r < P_i \tag{4.31}$$

Figure 4.10 demonstrates how the sampler favours the label with lowest interaction energy, i.e., maximum *a posteriori* probability. Assuming that the local configuration of site s at iteration t remains the same as in iteration $t-1$, that is to say $U(\lambda_s, \lambda_{\mathcal{N}_s})$ remains unchanged in iteration $t-1$ and

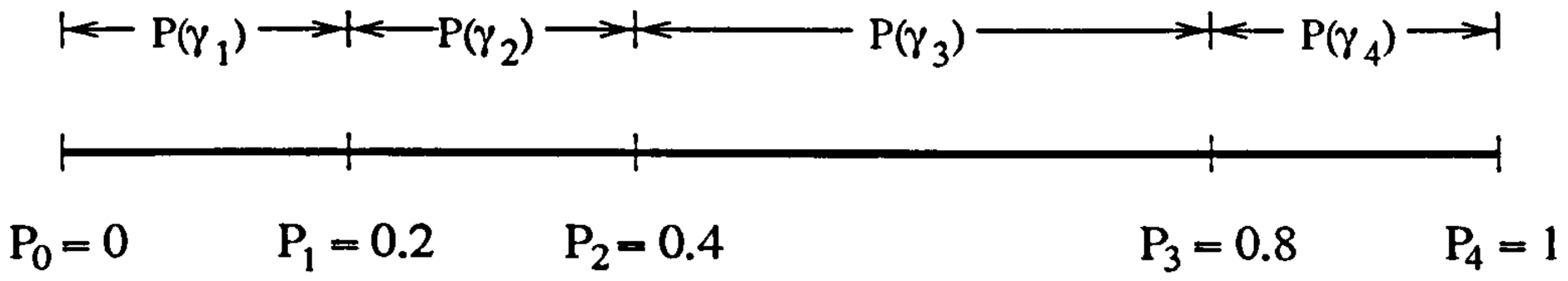


Figure 4.9: Probabilities of assigning four labels, $\gamma_1 - \gamma_4$, to a site.

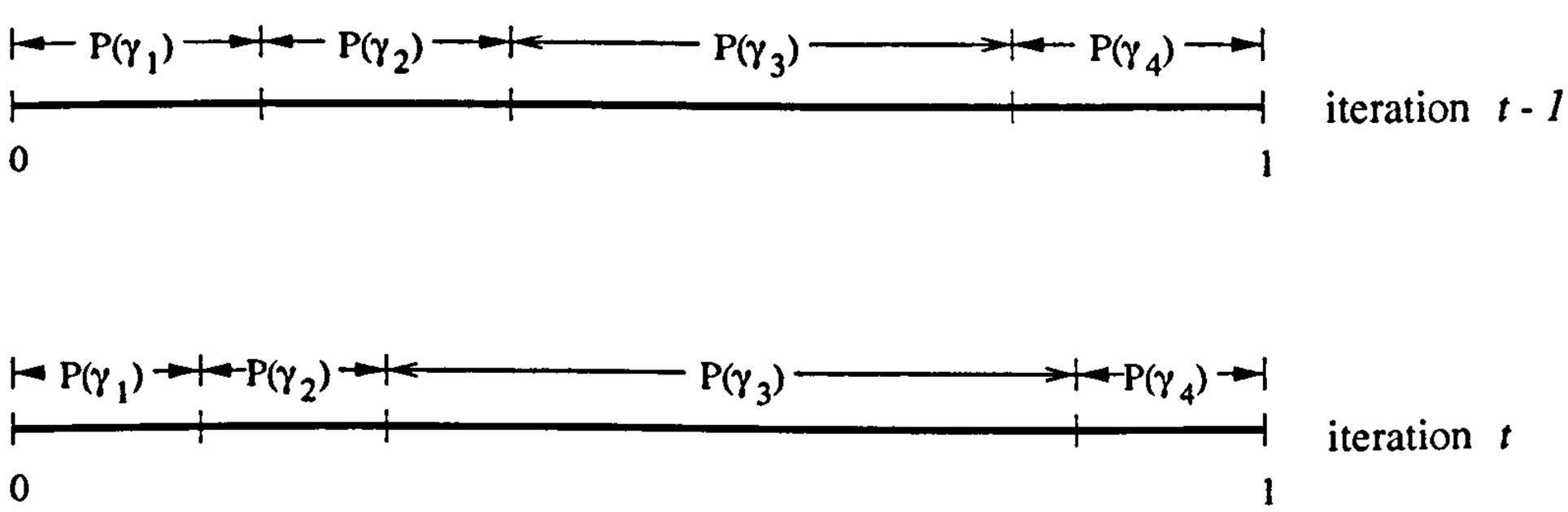


Figure 4.10: Probabilities of assigning four labels to a site at two consecutive iterations. Note $P(\lambda_s = \gamma_i|\lambda_{\mathcal{N}_s})$ has been shortened as $P(\gamma_i)$

t, then according to Equation (3.14), as the temperature T goes down (T decreases monotonically with the number of iterations), the probability of the label with lowest interaction energy in iteration t will become higher than in iteration $t-1$ while the probabilities of the other labels will decrease. So, as the temperature T keeps dropping, the probability of the label with lowest interaction energy will keep increasing and eventually the algorithm will settle in a configuration where no more changes occur. If there is no label change after a full iteration, we say that the algorithm has converged.

Since the algorithm might converge to a local minimum configuration, in our work, we allow the algorithm to continue for a few iterations after convergence so as to give it a chance to climb out of the local minimum. If no more changes occur throughout those iterations, the segmentation at the current resolution level is done.

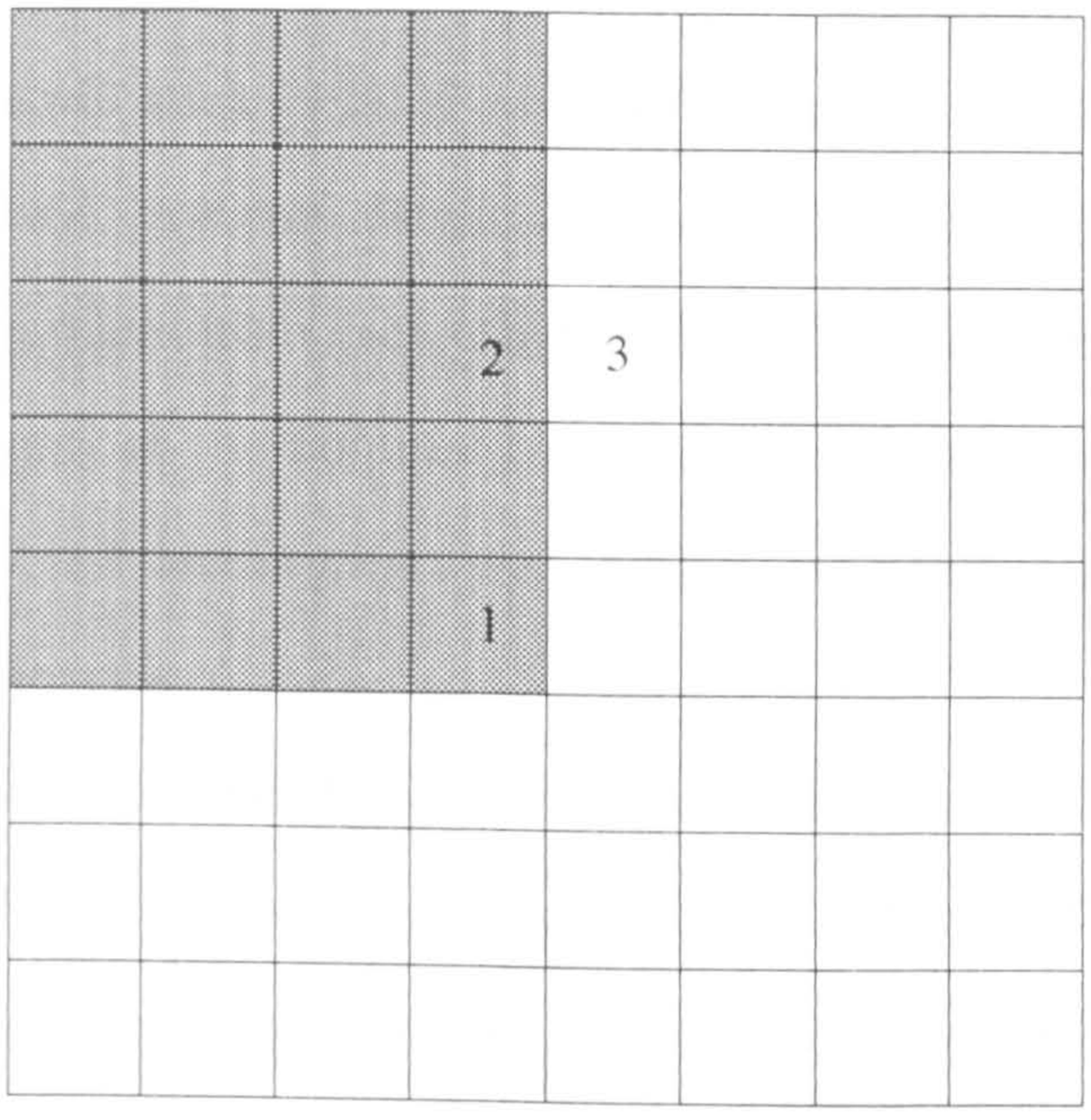


Figure 4.11: A homogeneous region being wrongly segmented into two regions.

4.2.3 Breaking Deadlocks

Depending on the type of neighbourhood imposed, sometimes a homogeneous region does get segmented into different regions by mistake. For example, with a 4-neighbourhood as adopted in this work, Figure 4.11 shows a homogeneous region being segmented into two regions. One problem occurs at site 1 in Figure 4.11 where two of its 4-neighbours got labelled *gray* while the other two were labelled *white*. Since all of them belong to the same class, the two *gray* sites exert about the same amount of attraction force as the two *white* ones do on site 1 and delay the convergence seriously. This situation may cause the algorithm to oscillate at site 1. Another problem occurs at the other sites along the false boundary. Taking sites 2 and 3 for example, when site 2 is visited, three of the 4-neighbours are labelled *gray* while only one is *white*. Therefore, the probabilities of labelling site 2 *gray* will keep

increasing. Similarly, site 3 is next to three *white* neighbours and only one *gray* one, thus it will stick to label *white* with high probability. We call the two aforementioned problems *deadlocks* because both sibling regions try to differentiate themselves from each other without compromise. Deadlocks are most likely to occur at the nominal top level in this work because, for each site, there is no *soft binding* between children and father sites to prevent internal inconsistencies.

To break deadlocks, when 90% of the sites at a level have their maximum a posteriori probabilities of assigning *any* label greater than 0.95, the algorithm detects the occurrence of deadlocks. Should there be any deadlocks, the region with smaller population is re-labelled with the label of the other. To detect deadlocks, for each pair of neighbouring regions, the average of *intra-region* and *inter-region* feature measurement $X_{ss'm}$ with highest weighting factor α_m, σ^2_{wm} and σ^2_{om}, are calculated to see if the following Inequality (4.32) holds.

$$|1 - \frac{\sigma^2_w}{\sigma^2_o}| < 0.1 \tag{4.32}$$

If Inequality (4.32) holds, it is very likely that there is a deadlock between the two neighbouring regions.

4.2.4 Data Propagation Across Resolutions

Once the algorithm converges at a specific level $k-1$, the information gathered at this level is propagated down to the next level (level k). The information to be gathered upon convergence is

- Final configuration. This information is employed to:

1. Calculate the mean gray level of each segmented region which, in turn, is to be used at the immediate level below in the calculation of X_{sf} in Equation (4.25) and potential $V_f(\lambda_s, \lambda_f, X_{sf})$, σ^2_{wf} and σ^2_{of} in Equation (4.24).

2. Assign the initial configuration of the next level.

- Texture boundaries. These are utilised to calculate $P(\lambda_s = \lambda_f)$ and $P(\lambda_s \neq \lambda_f)$ in Equation (4.23) and (4.24), respectively. We denote

$$P(\lambda_s = \lambda_f) = 1 - \rho_k^d \tag{4.33}$$

where d is the shortest distance between site s and a site having a different class, i.e., it represents distance to the *boundary propagated from the upper level.* For example, Figure 4.12(a) shows an image segmented into two regions. Figure 4.12(b) shows the initial configuration and coarse boundary at level k inherited from level $k-1$. In Figure 4.12(b), site s_1 is right next to the propagated boundary, so $d = \sqrt{1^2 + 0} = 1$. For site s_2, $d = \sqrt{2^2 + 0} = 2$, and for s_3, $d = \sqrt{2^2 + 1^2} = \sqrt{5}$. Our argument is that at higher level, the class resolution is high while the position resolution is low, so the classification of sites farther away from the detected boundary is more reliable than the sites closer to the boundary. Based on this reasoning we expect that if a site is further away from the propagated boundary, the possibility of the site taking the same label as its father is higher, and vice versa. ρ_k in Equation (4.33) is a constant less than 1. By varying ρ_k, it is possible to accommodate the appearance of regions too small to register at the largest scales.

While, at the nominal top level, $P(\lambda_s = \lambda_{s'})$ and $P(\lambda_s \neq \lambda_{s'})$ are recalculated at the beginning of each new iteration according to the

population of the assigned labels, at each level other than the nominal top level, their values are calculated according to Equation (4.34) at the first iteration and remain constant throughout the optimisation process at that level.

$$P(\lambda_s = \lambda_{s'}) = 1 - \frac{1}{2}(\rho_k^{d_s} + \rho_k^{d_{s'}}) \tag{4.34}$$

where d_s and $d_{s'}$ are the shortest distances from sites s and s' to the propagated boundary, respectively. ρ_k is a variable less than 1 which depends on level k. The reason Equation (4.34) is thus defined is twofold. First, the probability that a pair of sites farther away from the boundary belong to the same class is higher than the probability of the pairs closer to the boundary, so they are functions of distance. Secondly, the lower the level in the MFT structure, i.e., the smaller the block size is, the higher is the probability that neighbouring sites belong to the same class. Therefore, ρ_k should be given a lower value to reflect this situation at lower level. In the rest of this work, all the experiments are conducted on images of size 256×256 and the algorithm starts at level 3 (block size 64×64) and ends at level 6 (block size 8×8), so we denote

$$\rho_k = 0.25 - 0.05(k-4), \quad k = 4,5,6 \tag{4.35}$$

This empirical formula reflects the fact that a boundary site will usually have 3 neighbours in the same class, i.e. boundaries are locally straight lines.

In summary, the algorithm can be described in short as shown in Figure 4.13.

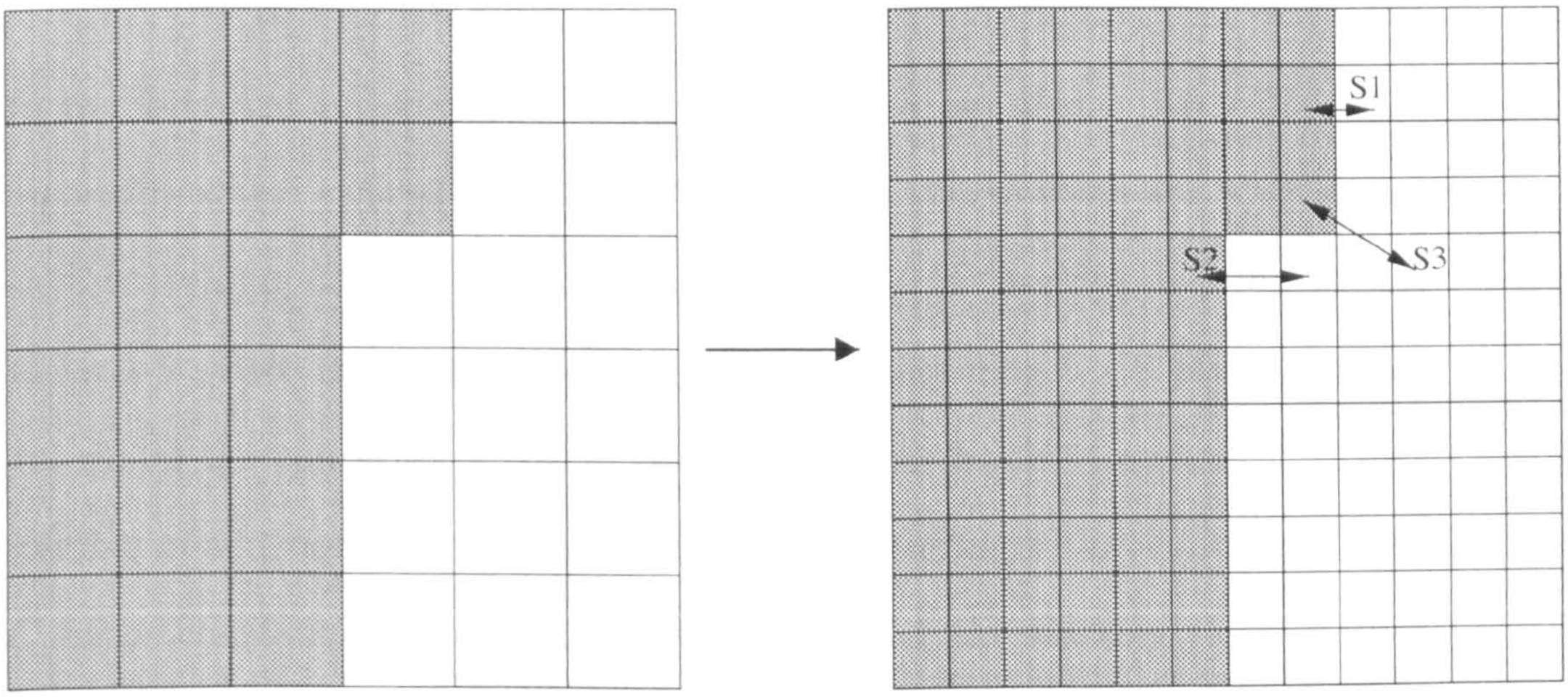


Figure 4.12: Distance from the propagated boundary. (a) Boundary detected at level k-1 (b) Boundary propagated from level k-1 down to level k.

1. high-pass filter the input image

2. current_level k = nominal_top_level

3. extract texture feature measurements $X_{ss'm}$

3.1 perform multiresolution Fourier Transform

3.2 extract texture features from each sites

4. Segment current level using Gibbs Sampling scheme

4.1 for all sites, label each site based on MAP

4.2 if converge then continue; else go to step 4.1

5. if current_level = nominal_bottom_level then go to step 7;

else propagate information to next level (level k + 1)

6. current level k = k + 1; go to step 3

7. stop

Figure 4.13: Region-based texture segmentation algorithm.

4.3 Experiments

In the following experiments, all the images used are of size 256×256, the algorithm starts at level 3 of the MFT structure, where there are 8×8 sites, each containing 64×64 pixels (because of the 50 % overlap) and completes at level 6 where there are 64×64 sites, each containing 8×8 pixels. Table 4.2 shows the constants I and C used at different levels. Constant I is the number of additional iterations carried out after the algorithm converges, and C is the constant in Equation 3.17 which decides the starting temperature of the algorithm: the higher the level, the larger the constant. The reason for this is that the coarser segmentation results at the higher level will be propagated down to the lower levels; we want to obtain the coarser results as accurately as possible, as long as we can afford the computational price. Since the population of the site at level k is 4 times less than that at level $k+1$, i.e., the computational load at level k is also 4 times less than that at level $k+1$, we can afford, at higher levels, to start at higher temperature so as to provide the algorithm a more moderate annealing schedule and to allow the algorithm higher value of constant I in order to give it more chance to climb out of local minimum configurations.

Figure 4.14 to Figure 4.18 show the segmentation results of images consisting of different textural regions separated by different boundaries. The refinement of the segmentation results across levels can be seen clearly. These are quantified by listing the error rates, percentage of misclassification, in Table 4.2. The numbers of iterations per pixel are listed in Table 4.3. Note that for each experiment, there is a significant drop in the number of iterations per pixel from level 3 to 4 because of the big value changes of constant C

Table 4.2: Error rate of segmentation results

level k	C	I	Error rate (%)				
			Image I	*Image II*	*Image III*	*Image IV*	*Image V*
3	8	3	7.053	13.075	11.464	6.892	2.600
4	6	1	1.640	5.080	1.807	5.739	2.173
5	4	1	1.341	2.278	1.312	3.795	1.401
6	3	1	0.993	1.964	0.876	3.185	1.385

Table 4.3: Number of iterations per pixel when the algorithm converges.

level k	Number of iterations per pixel				
	Image I	*Image II*	*Image III*	*Image IV*	*Image V*
3	0.189	0.340	0.186	0.202	0.182
4	0.074	0.129	0.066	0.125	0.058
5	0.349	0.562	0.531	0.594	0.250
6	2.191	2.626	2.499	2.620	2.064
Total	2.803	3.657	3.282	3.547	2.554

and I. However, from level 4 downward, the number of iterations per pixel increases across levels because of the larger population of image blocks at lower level and the fixing of constant I, despite the fact that constant C is lowered (i.e., the lower the level, the lower the temperature the algorithm starts at). The most significant feature of these results is the extremely low number of iterations/pixel. This is a considerable improvement over pixel based MRF methods [36].

The two textures of *Image I* in Figure 4.14, have distinctive mean gray level and possess structural properties with relatively regular spatial arrangement. Thus both the statistical and structural components of the feature set have their contribution to the discriminating task. Five regions are detected,

with three small false ones along the real boundary at level three. The reason for the three false ones to appear is that those small regions contain equivalent amounts of texture from different classes and the texture combination makes them differ significantly from both of the two classes. However, the three false regions are eliminated at level 4 and the boundary estimate is significantly improved (see Table 4.2).

The two textures of *Image II* in Figure 4.15, show prominent structural properties but low gray level contrast, thus we can expect that the success of the algorithm relies mainly on the two structural measurements $X_{ss'3}$ and $X_{ss'4}$ rather than on the two statistical measurements $X_{ss'1}$ and $X_{ss'2}$.

Image III in Figure 4.16 contains one statistical texture and a deterministic one. In this case all the 4 measurements all give their contributions. A ragged boundary is also employed to test the capability of the algorithm to detect such boundaries (see Figure 4.16(c) and (d)).

Figure 4.17 demonstrates the algorithm's robustness in detecting sharp boundary junctions, while Figure 4.18 demonstrates the algorithm's capability in detecting small regions. In Figure 4.18, the round region with 48 pixels in diameter centred at pixel $(137, 121)$ is segmented satisfactorily. However, if a smaller region of the same texture with 32 pixels in diameter is present in the image, it cannot be detected by applying the algorithm at level 3 because the window size is far larger than the textured region.

It is difficult to compare the feature extraction and segmentation approaches because there is no accepted standard set of textures on which to base a performance measure and some approaches perform segmentation without supervision, while most of the algorithms require human intervention. However, in order to show the robustness of our algorithm, we choose

the algorithms of Lu et al. [83], and Wilson and Spann [133] for comparison because their feature extraction methods and segmentation techniques were surveyed in Section 2.1.1 and 2.2.1, respectively, and both of their methods and ours were applied on similar textured images without supervision.

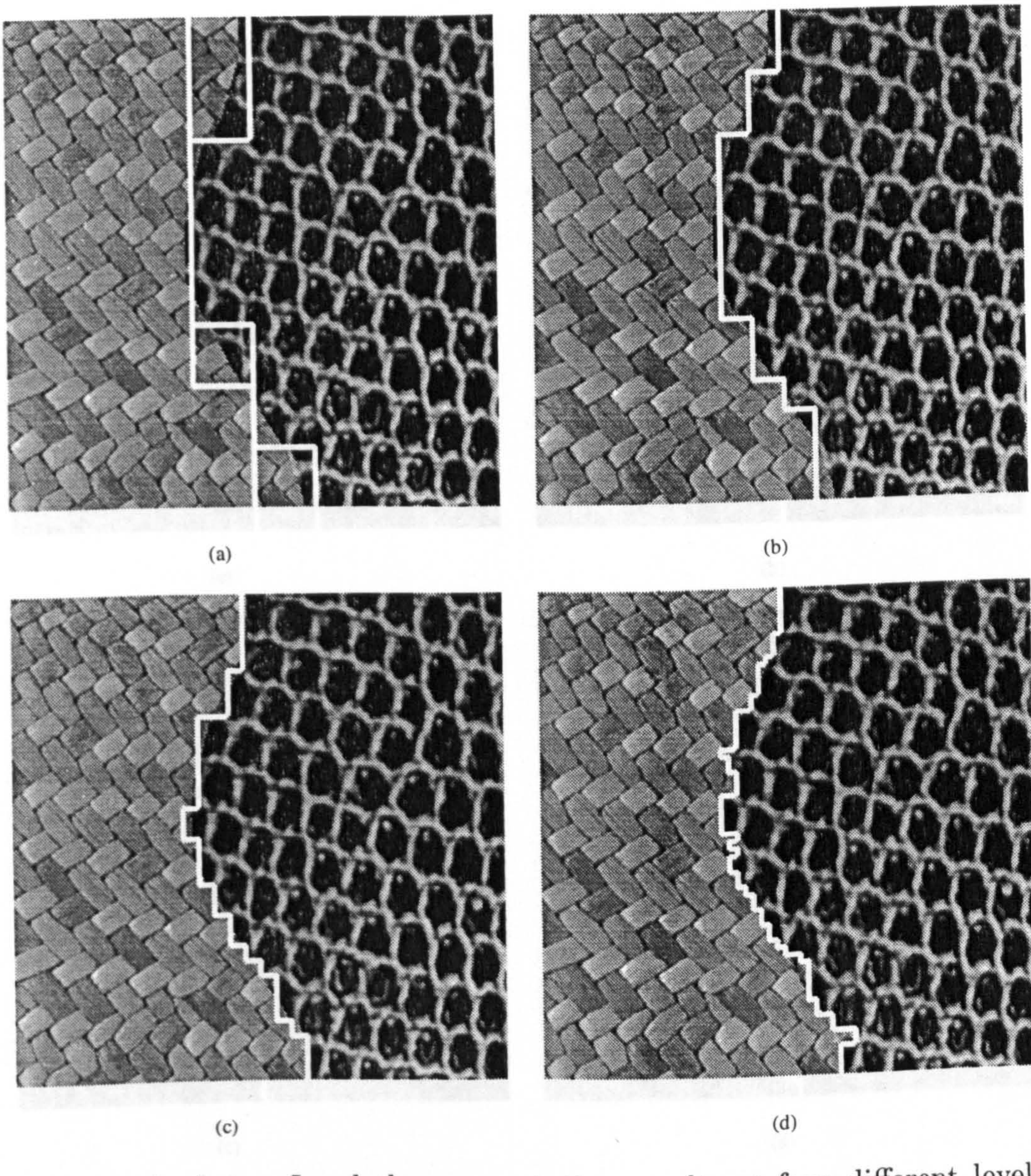
In [83], Lu et al. applied their algorithm which required two-scale wavelet decomposition to a textured image of size 256 × 256 pixels containing four equal-sized square textured regions. A classification error rate of 4.5% was reported. Comparing their result with ours of *Image IV* (Figure 4.17) in Table 4.2, we can see that on level 5, the error rate of our algorithm drops down to 3.8% which has outperformed Lu's algorithm. The error rate of our algorithm drops further down to 3.2% on level 6.

Wilson and Spann applied their algorithm to four textured images Figure 5.10—5.13 in [133], each consisting of two natural textures divided by a boundary of 256 pixel long. The resultant classification error per boundary point for each image was 1.8, 2.3, 2.9 and 2.5 (shown in Table 5.1 in [133]), respectively, which is equivalent to an error rate of 4.7%, 5.9%, 7.4% and 6.4%, respectively. Comparing to our results of *Image I*—III (Figure 4.14—4.16) shown in Table 4.2, each also consisting of two natural textures divided by a boundary of 256 pixel long, the worst one is that of *Image II* with an error rate at only 2.0% on level 6.

4.4 Conclusions

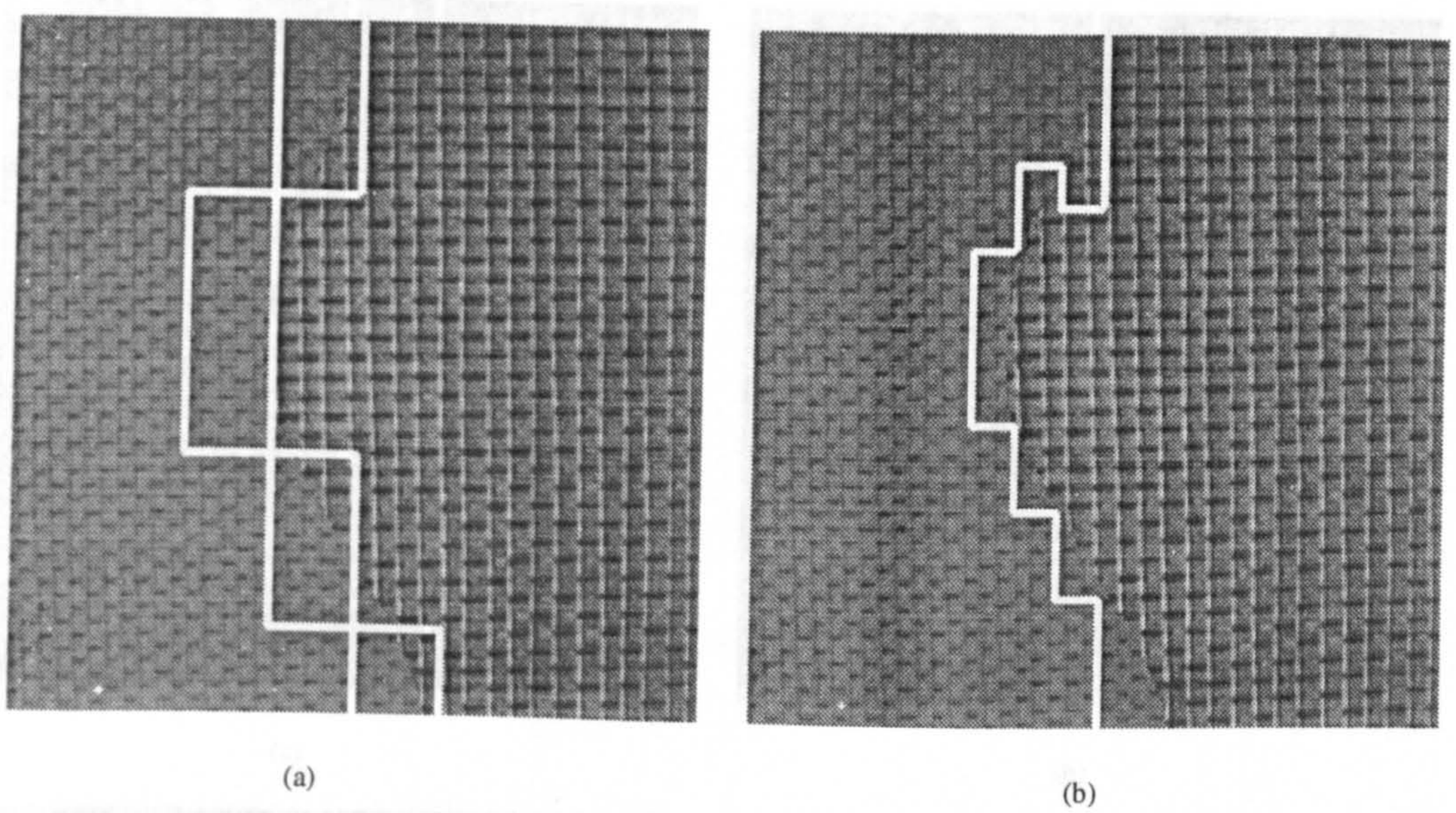
In this chapter, we have presented a novel approach to texture segmentation combining two important ideas: multiresolution MRF's to control the segmentation process and a two-component texture model, in which a deformable template is used to model the structural element of the texture and

the energy spectrum is used to capture the stochastic element. In effect, this separation of the classification model from the texture model creates a highly flexible and general segmentation tool. Moreover, the effectiveness of the method has been shown with results from examples using natural textures. The potential for the MMRF to overcome the drawbacks of using a single resolution, both in terms of accuracy and computation time, has been clearly demonstrated. Nevertheless, the blocking artifacts of this region-based algorithm calls for remedy because no attempt has been made to model the boundary explicitly, through a boundary or line process, although that is clearly feasible within the general MMRF framework [37]. We will propose a complementary boundary-based process in Chapter 5 to alleviate these effects and refine the boundary detected by the region-based process proposed in this chapter.



(a) (b) (c) (d)

Figure 4.14: *Image I* and the segmentation results at four different levels with estimated boundary superposed. a) Level 3, b) Level 4, c) Level 5, d) Level 6



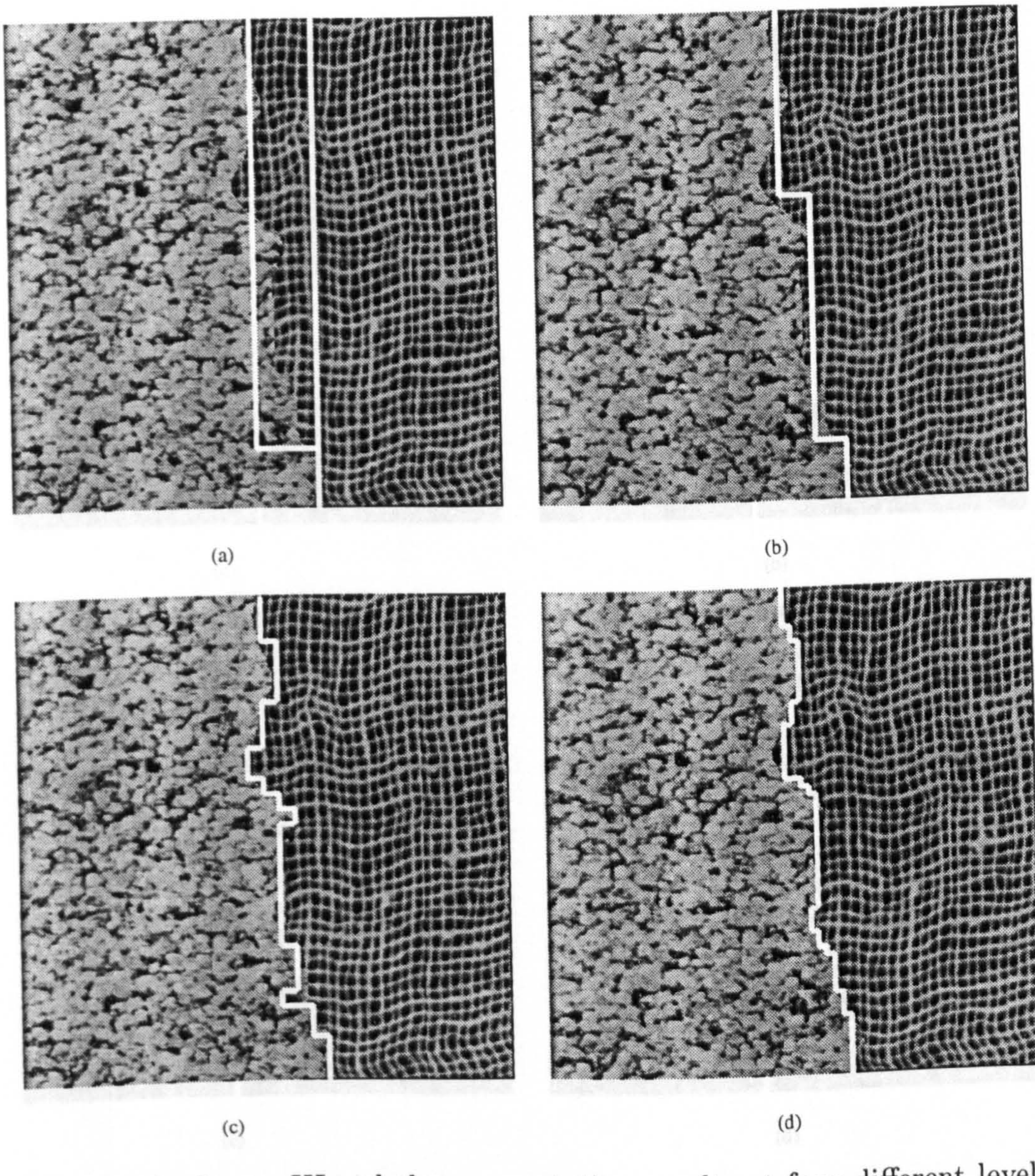
(a)

(b)

(c)

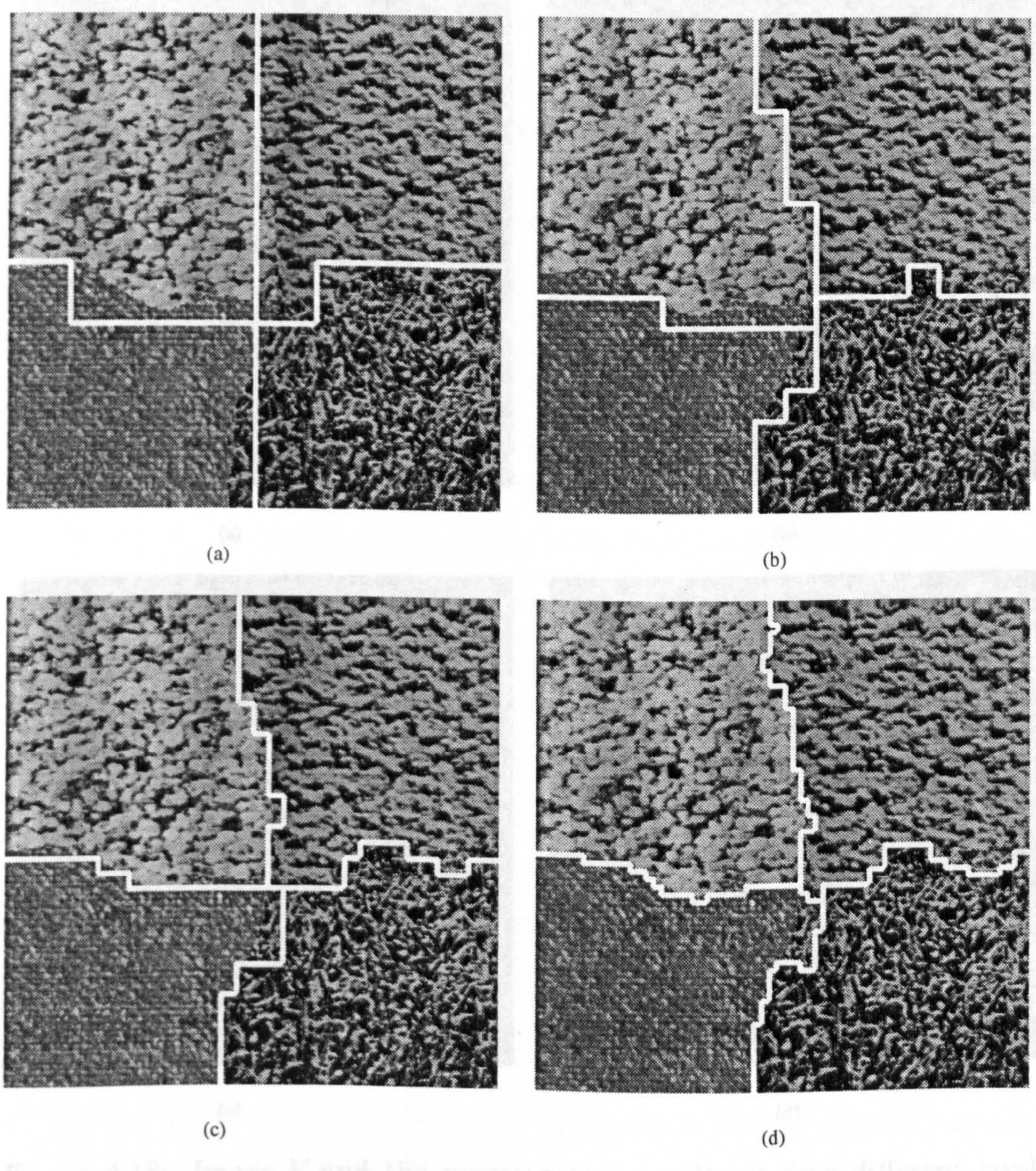
(d)

Figure 4.15: *Image II* and the segmentation results at four different levels with estimated boundary superposed. a) Level 3, b) Level 4, c) Level 5, d) Level 6



(a) (b) (c) (d)

Figure 4.16: *Image III* and the segmentation results at four different levels with estimated boundary superposed. a) Level 3, b) Level 4, c) Level 5, d) Level 6



(a) (b) (c) (d)

Figure 4.17: *Image IV* and the segmentation results at four different levels with estimated boundary superposed. a) Level 3, b) Level 4, c) Level 5, d) Level 6

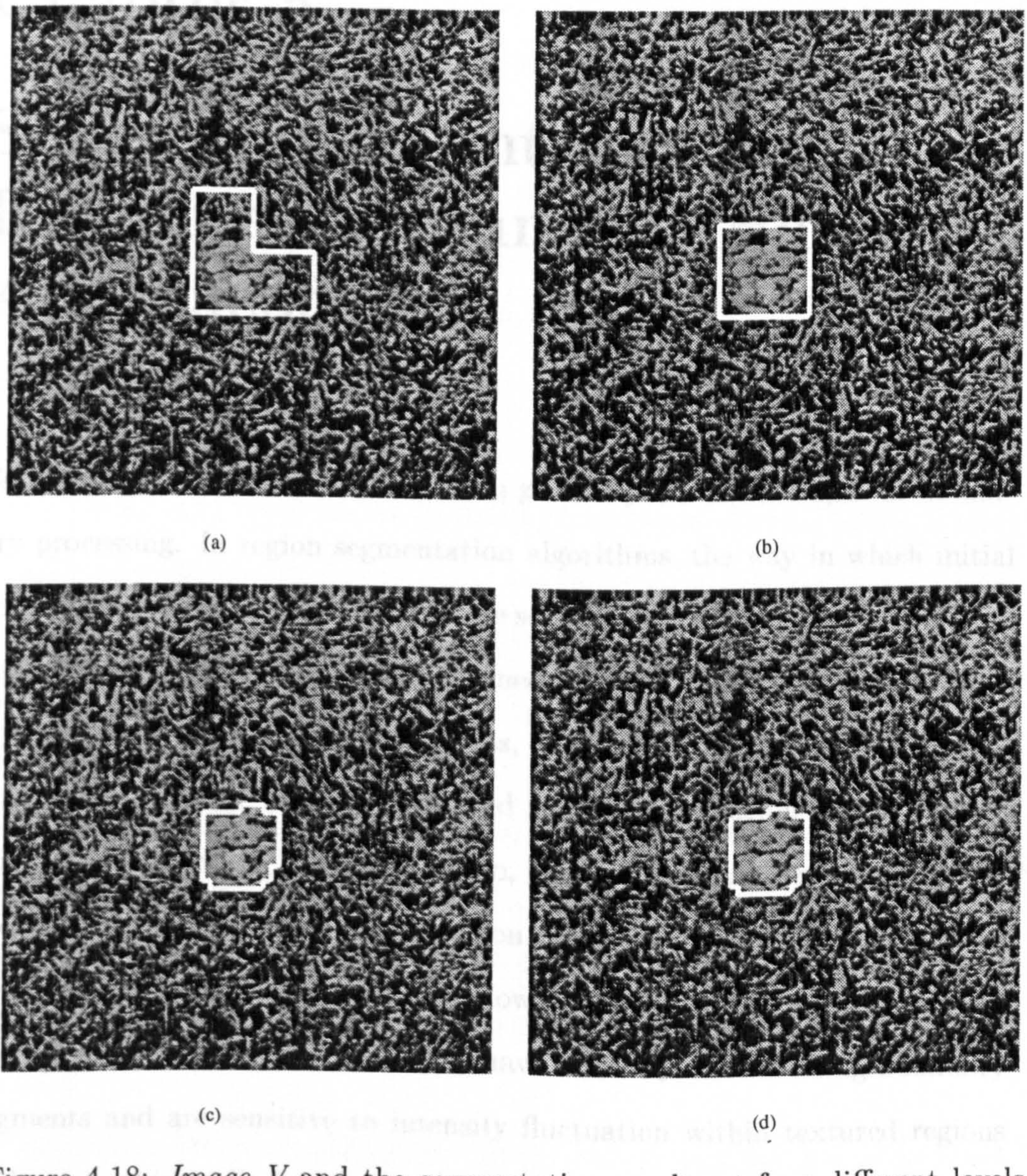


Figure 4.18: *Image V* and the segmentation results at four different levels with estimated boundary superposed. a) Level 3, b) Level 4, c) Level 5, d) Level 6

Chapter 5

Segmentation Integrating Region and Boundary Based Approaches

Conventional methods of segmentation generally use either region or boundary processing. In region segmentation algorithms, the way in which initial regions are formed, the initial seeds are selected, and the criteria for splitting and merging regions are usually defined *a priori*. If the process is non-ergodic, as in deterministic algorithms, then the segmentation result relies on the choice of the initial regions and the resultant region shapes depend on the chosen growth algorithm. Also, since region-based approaches seek localisation in class space, confidence on the boundary position is weakened by the use of a neighbourhood or window. Boundary based segmentation algorithms, on the other hand, usually have difficulty in connecting boundary segments and are sensitive to intensity fluctuation within textured regions and noise, which calls for post-processing by some form of smoothing or linking. It is clearly preferable to combine both approaches to overcome their inherent limitations.

In Chapter 4, we presented a method for segmenting textures using a Mul-

tiresolution Markov Random Field (MMRF) formulation, based on a novel texture model, which combines a structural component with a statistical one [52]. We showed in that chapter that MMRF models lead to a computationally efficient and robust segmentation. However, the algorithm makes use of only region information. The input image is divided into an array of square blocks, each of them treated as a basic unit/site during the segmentation process, so that the resulting boundary estimates were limited by the uncertainty associated with the size of the window used for sampling the image at each resolution. Upon completion of the algorithm at each resolution, the approximate boundary is drawn between blocks labelled differently. This approximation inevitably results in blocking artifacts shown in Figure 5.10—Figure 5.14. Although refinement at successive levels of finer resolution (smaller block size) does alleviate the uncertainty, the blocking artifacts still remain. Furthermore, since the initial configuration of a level, except the nominal top level, depends on the boundary information detected at its immediate ancestor level, the inaccuracy of the boundary approximation may be propagated. To overcome these shortcomings, we describe in this chapter a boundary process which is based on the same underlying MMRF framework as the region process introduced in Chapter 4 and complements the region process, giving improved boundary estimates at little extra computational cost.

Employing the same MFT-MRF framework at each level, the textured image is first segmented using the region process. Following the completion of the region-based phase, all of the image blocks on either side of the preliminary boundary are treated as potential boundary-containing blocks (PBCB). The orientation and the centroid of the boundary-segment contained in each

PBCB are estimated. The set of PBCB's is then modelled as a MRF and the interaction energy between each pair of neighbouring blocks is defined as a function of the 'distance' D between the centroids of the two boundary segments. Again stochastic relaxation simulating annealing is adopted to maximise the *a posteriori* probability for assigning *boundary-containing/non-boundary-containing* labels to the PBCB's. Once the algorithm converges, the centroids of the identified boundary blocks are connected to form a refined boundary, which is then propagated to the next resolution for further refinement.

5.1 Boundary-based Process

When the region process is completed and the boundary process assumes the refining task, all the blocks on either side of the preliminary boundary are treated as PBCB's because we are not certain through which of them the boundary actually goes. Figure 5.1(a) shows the segmentation result of a textured image using the region process only. The shaded blocks in Figure 5.1(b) are the PBCB's along the preliminary boundary.

The objectives of the boundary process are: first, to estimate the orientation and centroid of the boundary segment in each of the PBCB's so that the position of the boundary segments can be estimated, secondly, to classify the PBCB's into *boundary-containing* and *non-boundary-containing* classes based on the estimated orientations and centroids, so that the boundary can be more accurately approximated by connecting the boundary segments contained in the blocks labelled *boundary-containing* and the blocking artifacts of the segmentation result of the region process can be eliminated.

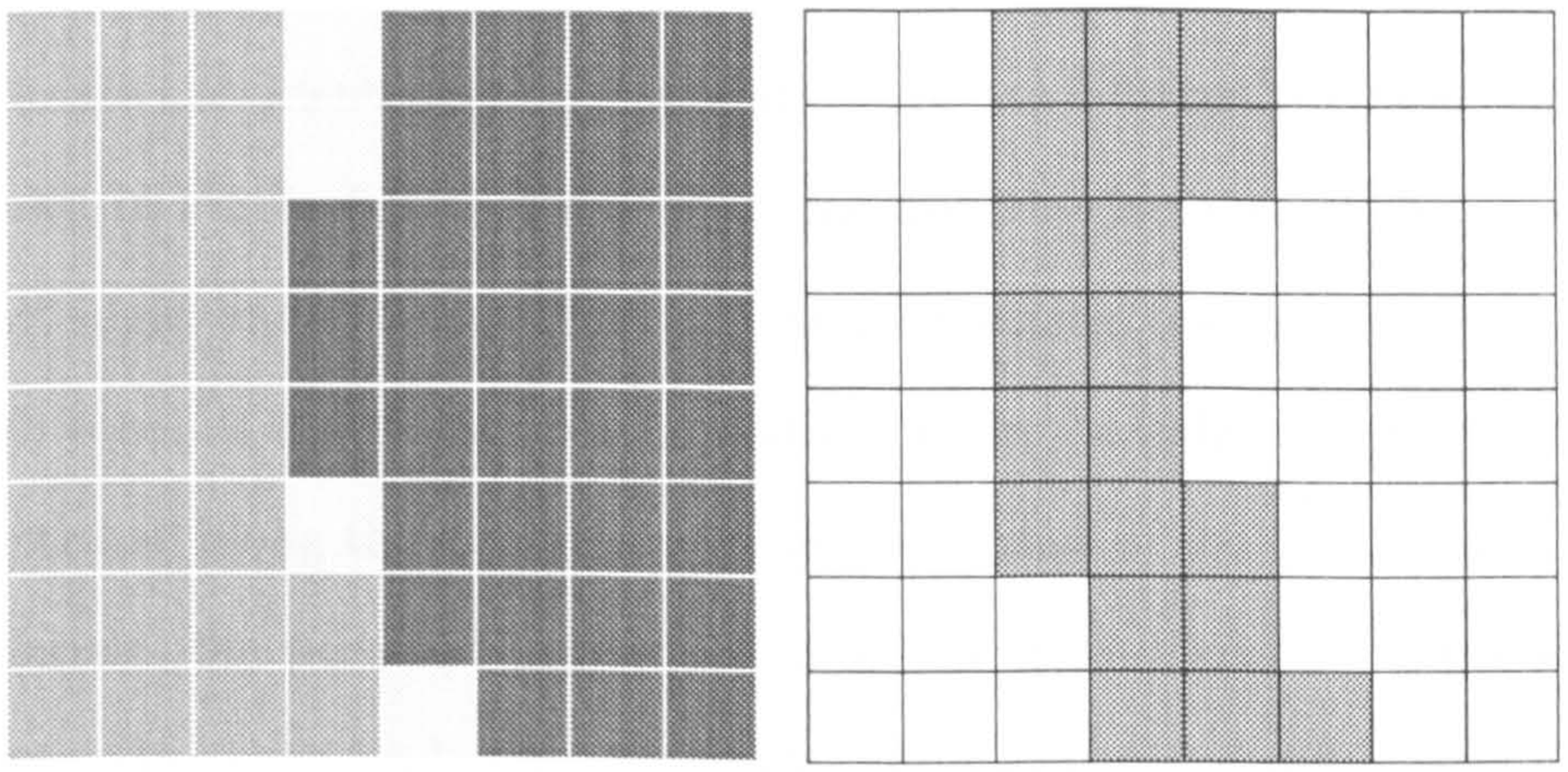


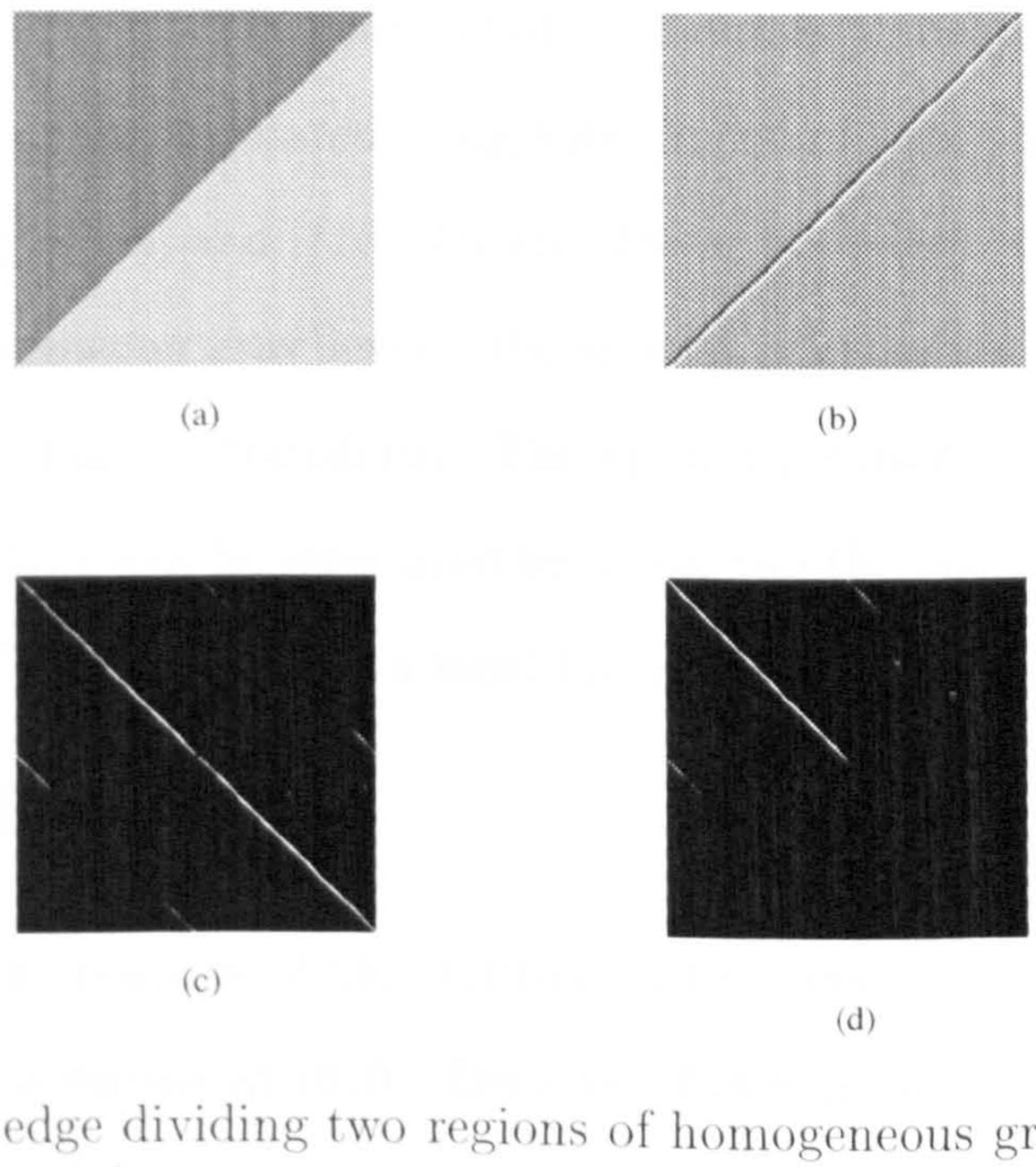
Figure 5.1: (a) Segmentation result using region process only. (b) Potential boundary-containing blocks (PBCB) along the estimated boundary represented by shaded blocks.

5.1.1 Estimating Boundary Orientation

For an edge between two regions with homogeneous gray levels in an image block, we can estimate its orientation by analysing its Fourier spectrum because the energy in the frequency domain concentrates along the orientation orthogonal to the orientation of the edge in spatial domain. For example, Figure 5.2(a) shows an edge segment between two regions of homogeneous gray level. In order to enhance the edge, Figure 5.2(a) is high-pass filtered as shown in Figure 5.2(b). Note that for the display purposes, the gray levels of the high-pass image in Figure 5.2(b) have been raised by 128 levels, so that all the pixels with gray level lower than 128 have a negative gray value in the real image while the pixels with gray level higher than 128 have positive gray value. Figure 5.2(c) shows the Fourier transform of the high-pass image. As expected, the energy is concentrated along the direction orthogonal to that of the edge in spatial domain.

There are a number of methods capable of estimating orientations of such

edges or boundary segments [76][129]. However, it is not feasible attempting to estimate the orientations of boundary segments between textured regions, as shown in Figure 5.3(a), using the same methods because of the presence of tiny edges and intensity fluctuations permeating textured regions. Although some of the energy corresponding to the boundary segment is still concentrated along the direction orthogonal to that of the boundary in spatial domain, the texture features with random orientations make the whole spectrum dispersed and thus overwhelm the signature of the energy corresponding to the real boundary segment. This is clearly shown in Figure 5.3(c).



(a) (b) (c) (d)

Figure 5.2: An edge dividing two regions of homogeneous gray level and its Fourier spectrum. a) the original image, b) the high-passed version of the original image, c) the Fourier spectrum of the high-passed version, d) the spectrum containing the Fourier coefficients in a half-plane.

We use information in texture feature space to estimate the orientation of boundary segments between textured regions. To start with, when the region

PAGE
MISSING
IN
ORIGINAL

-1	-2	-1
0	0	0
1	2	1

-1	0	1
-2	0	2
-1	0	1

Figure 5.4: A pair of Sobel operators for detecting intensity gradient. The left one is for detecting the amount of change in vertical direction while the right one is for the horizontal direction.

5.1.2 Estimating Boundary Centroid

Having estimated the orientations we still need more information to locate the boundary segments. The centroid of the boundary segment contained in each block is therefore calculated using a method similar to Calway's [15]. As described in [15][29][95] and [113], for an edge or boundary segment between regions of homogeneous gray levels, the spatial information is contained in the phase of the Fourier transform. The spatial position of the centroid of such a linear feature can be estimated by averaging the phase difference over all frequencies. The spectrum of a local linear feature may be written as [29]

$$\hat{f}(u,v) = \hat{l}(u,v)\, e^{-j(ux+vy)} \tag{5.1}$$

where (x, y) is the position of the centroid of the linear feature and $\hat{l}(u, v)$ is the spectrum of a feature at $(0, 0)$. Because of Hermitian symmetry only the coefficients in a half-plane as shown in Figure 5.2(d) have to be averaged.

The autocorrelation coefficients of the image spectrum in both the u and v dimension respectively are

$$\rho_u = \frac{\sum_{u,v\,\epsilon\Theta_\theta} \hat{f}(u,v)\, \hat{f}^*(u+u',v)}{\sum_{u,v\,\epsilon\Theta_\theta} |\hat{f}(u,v)|^2} \tag{5.2}$$

$$\rho_v = \frac{\sum_{u,v \in \Theta_\theta} \hat{f}(u,v)\, \hat{f}^*(u, v+v')}{\sum_{u,v \in \Theta_\theta} |\hat{f}(u,v)|^2} \tag{5.3}$$

where u' and v' are sampling intervals in u and v dimensions respectively. By substituting Equation (5.1) for both Equations (5.2) and (5.3), the estimate of the centroid position (x, y) can be given as

$$x_0 = \frac{Arg(\rho_u)}{u'} \tag{5.4}$$

$$y_0 = \frac{Arg(\rho_v)}{v'} \tag{5.5}$$

Since the sampling intervals in this work are $\frac{2\pi}{N}$, Equations (5.4) and (5.5) become

$$x_0 = \frac{N \cdot Arg(\rho_u)}{2\pi} \tag{5.6}$$

$$y_0 = \frac{N \cdot Arg(\rho_v)}{2\pi} \tag{5.7}$$

However, for a natural image with two textured regions as shown in Figure 5.3(a), the signature of the real boundary segment in the frequency domain as shown in Figure 5.3(c) is 'drowned' by the signatures of the texture fluctuation and tiny edges. Using Calway's method without any modification is not sufficient to achieve the goal of detecting the centroid. Thus, instead of taking into account all the Fourier coefficients in one half plane of the spectrum, we use only those within a strip along the orientation estimated using the method described in Section 5.1.1 as shown in Figure 5.3(e). This reduces the influence of texture fluctuations and separates the texture boundary from the texture features.

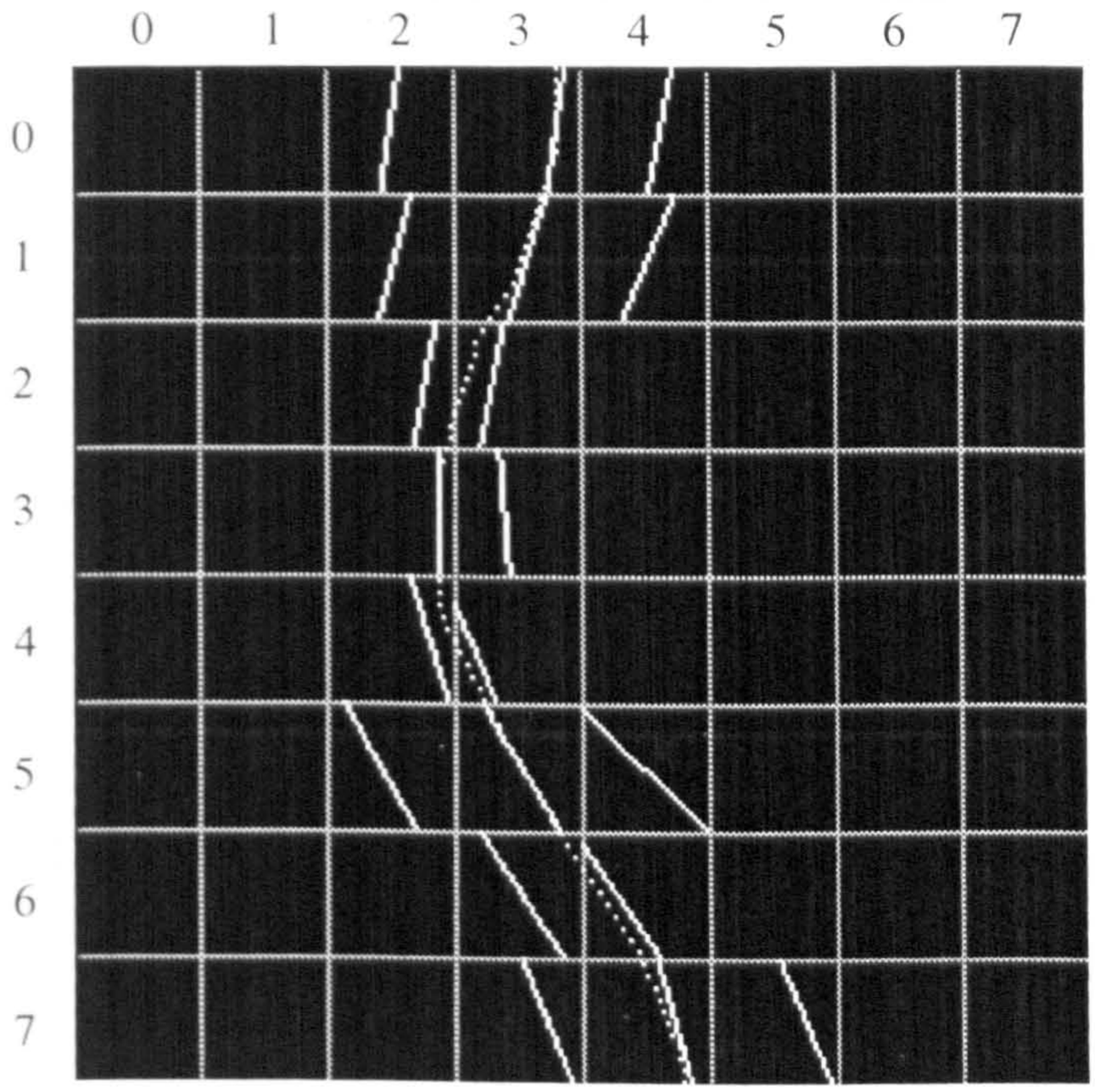


Figure 5.5: Potential boundary-containing blocks of Figure 4.14 with the estimated boundary segments drawn. The real boundary is also drawn for comparison

5.1.3 Classification

Once the orientations and centroids of the PBCB's are calculated, we want to classify them into *boundary-containing* and *non-boundary-containing* classes based on their associated orientations and centroids. Figure 5.5 illustrates the PBCB's of Figure 4.14 after the execution of region process at level 3, with their boundary segments drawn according to estimated orientations and centroids. The real boundary of the image is drawn as a dashed curve for comparison.

To classify them, the set of the PBCB' is modelled as a Markov Random Field and the same MRF framework underlying the region process is employed to optimise the classification. However, only the 4-neighbours which

are also PBCB's on the same level are included in the neighbourhood system, i.e.

$$\mathcal{N}_s = \{(i+i', j+j') | \quad (i,j) \text{ are the coordinates of site } s; \\ i', j' = \pm 1, (i+i', j+j') \text{ is a PBCB}\} \tag{5.8}$$

The clique system adopted is still $\mathcal{C} = \mathcal{C}_2$ (cf Chapter 4) and the label set Γ has only two elements, i.e. $\Gamma = \{boundary-containing, non-boundary-containing\}$.

Definition of Interaction Energy To define interaction energy, the clique potential has to be specified first. In this work, the potential between a site s and each of its neighbours s' is defined as a function of λ_s, $\lambda_{s'}$ and a 'distance' measure D between the estimated boundary segments. The 'distance' D is demonstrated in Figure 5.6 and defined as

$$D = \|\vec{l}\|(sin(\theta_1) + sin(\theta_2)), \ 0 \leq \theta_1, \theta_1 < \pi/2 \quad . \tag{5.9}$$

where $\vec{l}$ is the vector joining the centroids of the boundary segments within the two blocks, and θ_1 is the angle between $\vec{l}$ and one of the boundary segments and θ_2 is the angle between $\vec{l}$ and the other segment. From Figure 5.6, we can see that the more smoothly aligned a boundary segment pair, the smaller the θ_1 and θ_2, therefore, the shorter the 'distance' between them. If this is the case, both blocks are more likely to be boundary containing, a smaller cost (potential) is imposed to encourage labelling both blocks *boundary-containing*. If the 'distance' is relatively large, it is less likely that *both* sites are boundary containing, and a higher cost should be imposed to discourage labelling both blocks *boundary-containing*. This is to say that the potential energy $V(\lambda_s, \lambda_{s'}, D)$ is proportional to 'distance' D. For example,

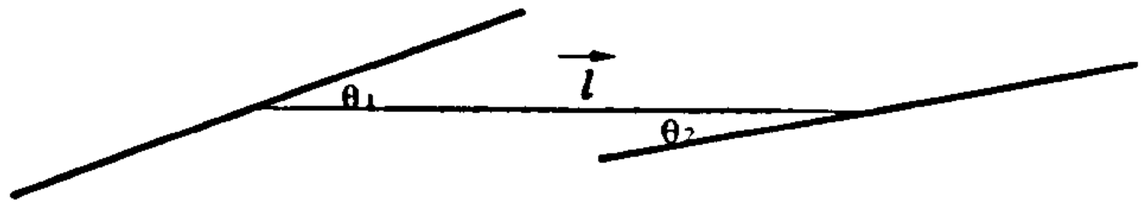


Figure 5.6: 'Distance' between boundary segments

in Figure 5.5 , $\|\vec{l}\|$ between blocks $(0,3)$ and $(0,4)$ is about the same as that between blocks $(0,3)$ and $(1,3)$. However, θ_1 and θ_2 associating blocks $(0,3)$ and $(0,4)$ are relatively close to $\frac{\pi}{2}$ while θ_1 and θ_2 associating blocks $(0,3)$ and $(1,3)$ are relatively close to 0, thus the distance D between $(0,3)$ and $(0,4)$ is much longer than that between $(0,3)$ and $(1,3)$. Therefore, we can expect that blocks $(0,3)$ and $(1,3)$ are more likely to be labelled *boundary-containing* while block $(0,4)$ is to be labelled *non-boundary-containing*. Based on the above argument, the potential between sites s and s' is defined, in a fashion similar to Equation (4.23), as following

$$V(\lambda_s, \lambda_{s'}, D) = \begin{cases} \ln\left(\frac{P(\lambda_s \neq \lambda_{s'})}{\sigma_o}\right) - \frac{D^2}{2\sigma_o^2} & , if\ \lambda_s = \lambda_{s'} \\ \ln\left(\frac{P(\lambda_s = \lambda_{s'})}{\sigma_w}\right) - \frac{D^2}{2\sigma_w^2} & , if\ \lambda_s \neq \lambda_{s'} \end{cases} \tag{5.10}$$

where σ_w and σ_o in Equation (5.10) are defined respectively as

$$\sigma_w^2 = \text{the average of the smallest 50 \% of square distances } (D^2) \tag{5.11}$$

$$\sigma_o^2 = \text{the average of the largest 50 \% of square distances } (D^2) \tag{5.12}$$

The reason we take 50% of the distances for calculating σ_w and σ_o is because about 50% of the PBCB's are *boundary-containing* and *non-boundary-containing*, respectively. Based on the same reason, letting $P(\lambda_s = \lambda_{s'}) = P(\lambda_s \neq \lambda_{s'})$ in Equation (5.10) is a reasonable choice and is not likely to affect the final labels significantly.

Since a strong gradient indicates the presence of a boundary segment with high probability, we also make use of the magnitude of the gradients

as a measure to classify PCBC's. If the gradient magnitude of a block is stronger than its neighbour *on the other side* of the preliminary boundary, it is more likely that the boundary goes through this block rather than the other. So in this case, we add a negative cost $\mathcal{E}$ to the total interaction energy to encourage the assigning of *boundary-containing* label to the block. Otherwise $\mathcal{E}$ is set to 0.

Treating the four image borders as part of region boundaries, we can impose a fundamental constraint on the algorithm that *a boundary is continuous without any disruptions in between boundary-containing blocks.* This constraint can be simply encoded into the interaction energy function by imposing that at least one of the two PBCB's next to the preliminary boundary segment detected in the region process should be labelled *boundary-containing.* If the assigning of a *non-boundary-containing* label violates the *boundary continuity* constraint a positive energy $\mathcal{P}$ is added to the interaction energy as a penalty, otherwise, $\mathcal{P} = 0$.

Based on the above discussions, the total interaction energy of each potential boundary-containing block with all its neighbours is therefore defined as

$$U(\lambda_s, \lambda_{\mathcal{N}_s}, D_{\mathcal{N}_s}) = \sum_{s' \in \mathcal{N}_s} V(\lambda_s, \lambda_{s'}, D) + \mathcal{E} + \mathcal{P} \tag{5.13}$$

where $D_{\mathcal{N}_s}$ represents the distances between the sites within the neighbourhood $\mathcal{N}_s$. Now let each block be a random variable taking values from the set $\Gamma = \{$*boundary-containing, non-boundary-containing*$\}$ and Λ be a set of label configurations of the sequence of potential block-containing blocks. A Gibbs distribution on the Λ-space can be defined in the same form as Equation 3.11, but using the energy defined in Equation 5.13 and the neighbourhood defined in Equation 5.8.

The task now is to find the maximum *a posteriori* (MAP) λ_s, i.e. $\lambda_s = arg\ max P(\lambda_s|\lambda_{\mathcal{N}_s})$. Due to ergodicity of Gibbs Sampler, initialising the labels of the PBCB's randomly will not affect the final configuration. However, this makes the algorithm slower to converge. As mentioned previously, one of the blocks next to a preliminary boundary segment must be a real boundary-containing block. Therefore, to reduce the computational cost, instead of initialising the labels randomly, for each block pair next to the preliminary boundary segment, we assign the label *boundary-containing* to the block with stronger gradient magnitude and *non-boundary-containing* to the other. When the simulated annealing process converges, the boundary is then connected. In this work we simply connect the centroids of the identified neighbouring boundary blocks to form the refined boundary.

A boundary is not only supposed to be continuous but also one site wide or the connecting operation will result in 'spaghetti' effects. Unfortunately, the requirement of one site width is not guaranteed to be met, therefore, a few spurious boundary-containing sites may appear when the algorithm converges. Observations suggest that a spurious boundary-containing block always appears as a *boundary terminator*. A boundary terminator can be defined as a site labelled *boundary-containing* which is isolated by non-PBCB's or blocks labelled *non-boundary-containing* in at least *five* directions within a 8-neighbourhood. The possible boundary terminators are shown in Figure 5.7. To meet the width requirement, these boundary terminators have to be turned off. The reason boundary terminators are not penalised during the sampling process is that this will not only penalise spurious boundary-containing blocks but also prevent a boundary from forming, because the real boundary-containing blocks next to the broken spots of the forming boundary

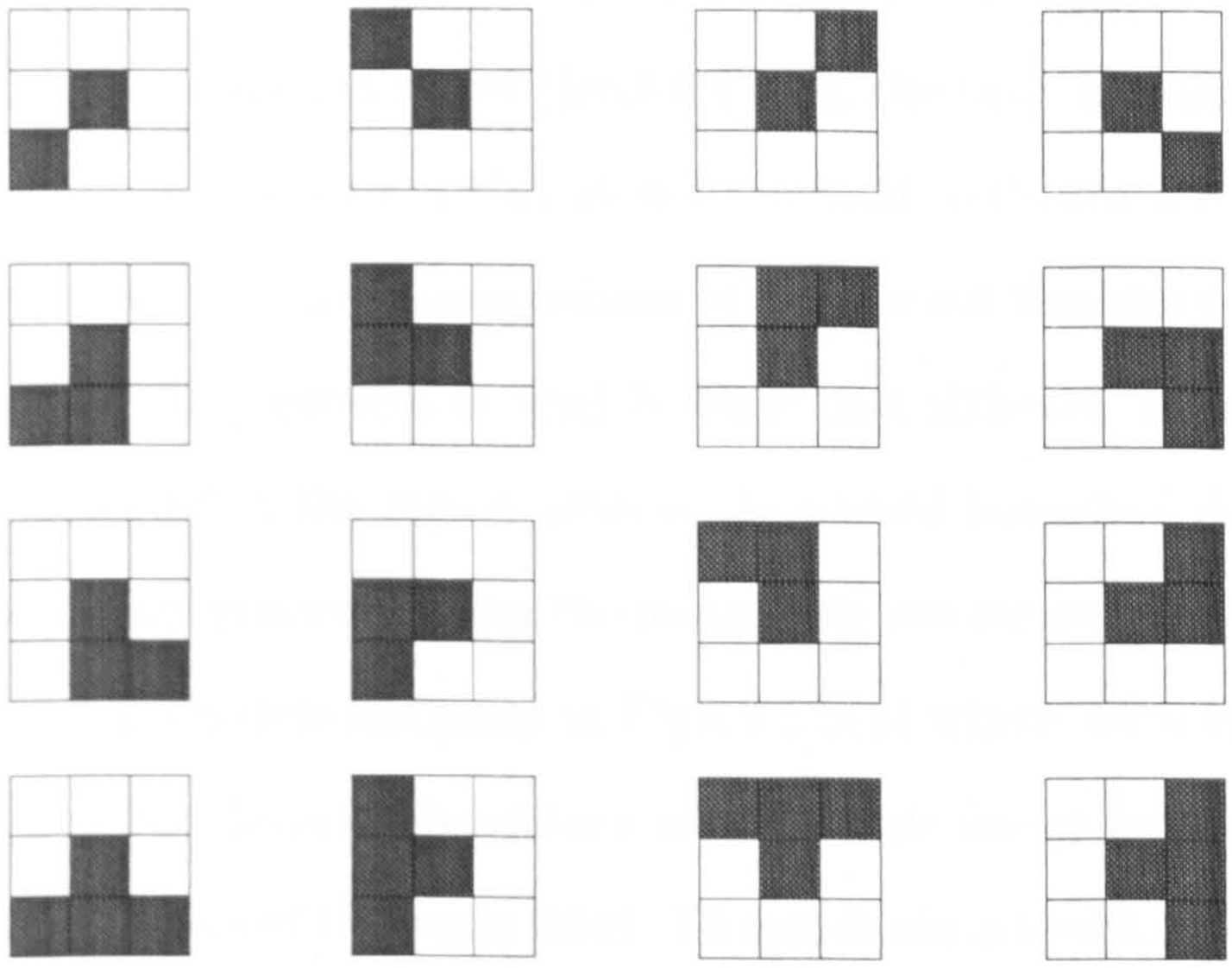


Figure 5.7: 12 types of *boundary terminator*. The terminator of each case is the central site within the 8-neighbourhood.

appear as boundary terminators too.

5.2 Data Propagation Across Resolutions

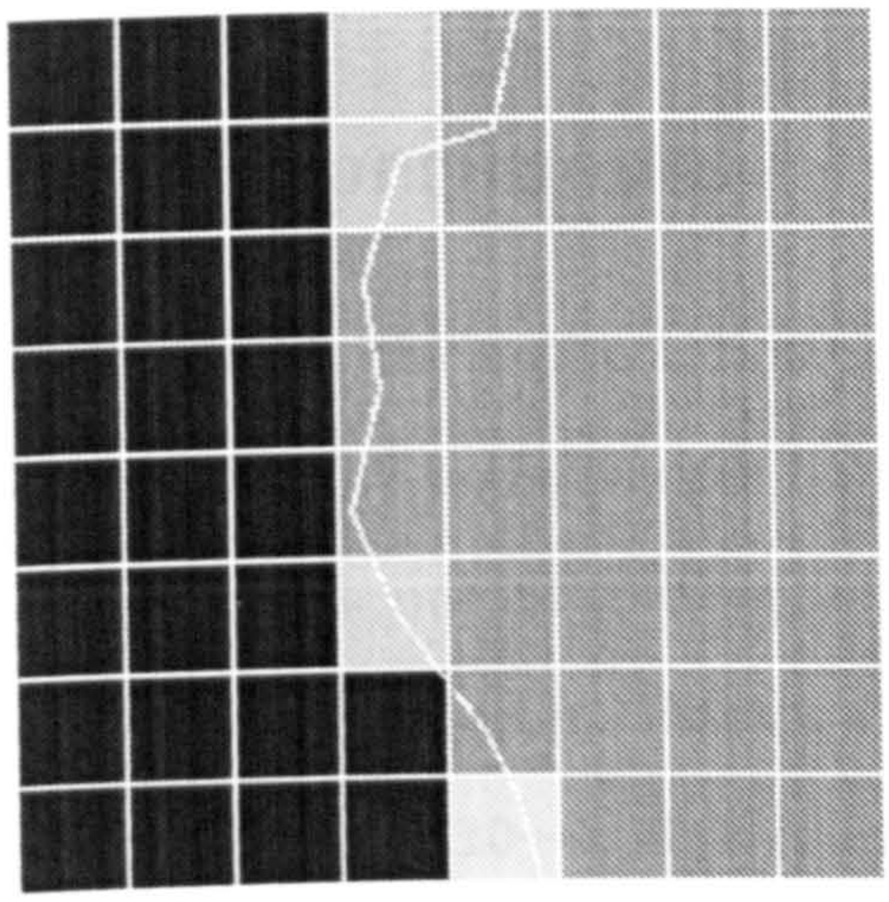
Once the real boundary blocks are identified, boundary information is propagated down to the next level and the algorithm repeats at the new level. In this model of processing, the estimates at level k act as a soft constraint on those at level $k+1$, penalising inconsistencies between adjacent scales. The information is employed to

1. Assign the initial configuration of the region process at the next level. Unlike their counterparts in the region process, the boundaries esti-

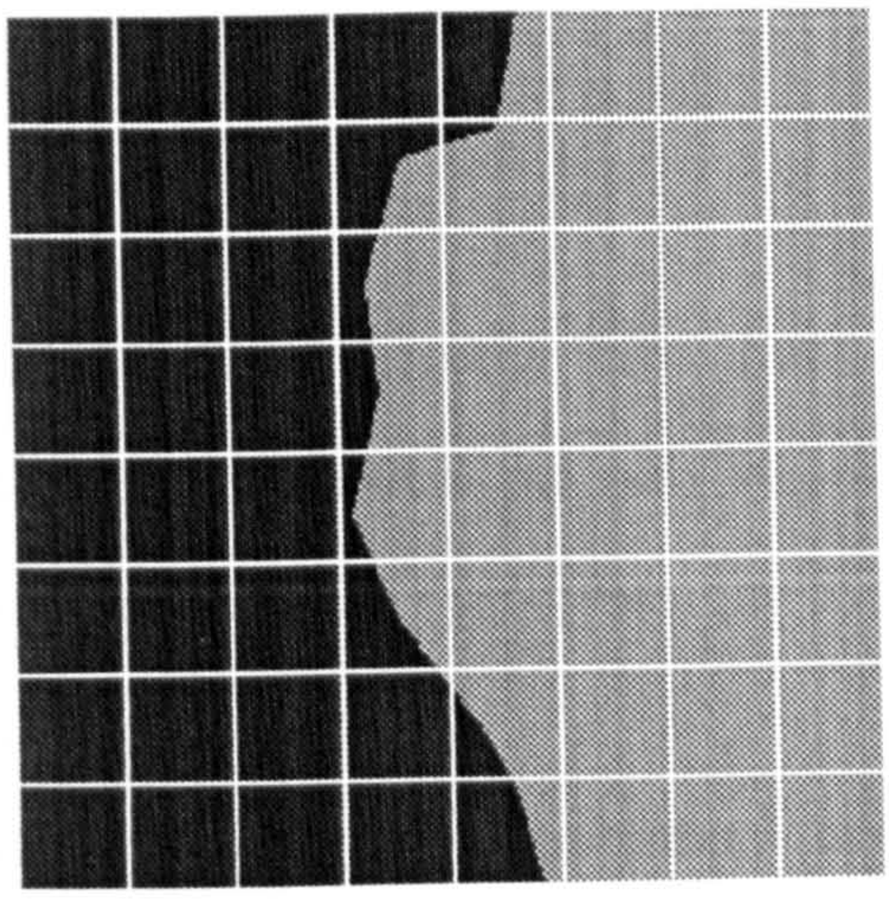
mated in the boundary process are curved, so they cannot be propagated down to the next level (level $k+1$) in the same manner as in the region process. Figure 5.8(a) shows the final configuration of the region process with the superposition of the curved boundary estimated in the boundary process at level 3. Note that although there are 5 regions detected in the region process the curved boundary detected by the boundary process divides the image into two regions — the real situation. This is demonstrated in Figure 5.8(b) where, with each region closed by the detected boundary and the four image borders, all the pixels are assigned the same label. The grids are added to demonstrate the size of the blocks/sites at level 3. Figure 5.8(c) shows the initial label configuration at level 4. Note that within each region, all the sites completely enclosed in that region are assigned the same label. However, for those sites which the boundary detected at the upper level goes through, their labels depend on what region most of their pixels belong to.

2. Calculate $P(\lambda_s = \lambda_f)$ and $P(\lambda_s \neq \lambda_f)$ in Equation 4.24 and $P(\lambda_s = \lambda_{s'})$ and $P(\lambda_s \neq \lambda_{s'})$ in Equation (4.23). By comparing Figure 4.12(b) and Figure 5.8(c), it is apparent that, for any levels other than the nominal top level, we can define $P(\lambda_s = \lambda_f)$ and $P(\lambda_s \neq \lambda_f)$ in Equation (4.24) in exactly the same way as we did in Equation (4.33), and $P(\lambda_s = \lambda_{s'})$ and $P(\lambda_s \neq \lambda_{s'})$ in Equation (4.23) in exactly the same way as we did in Equation (4.34).

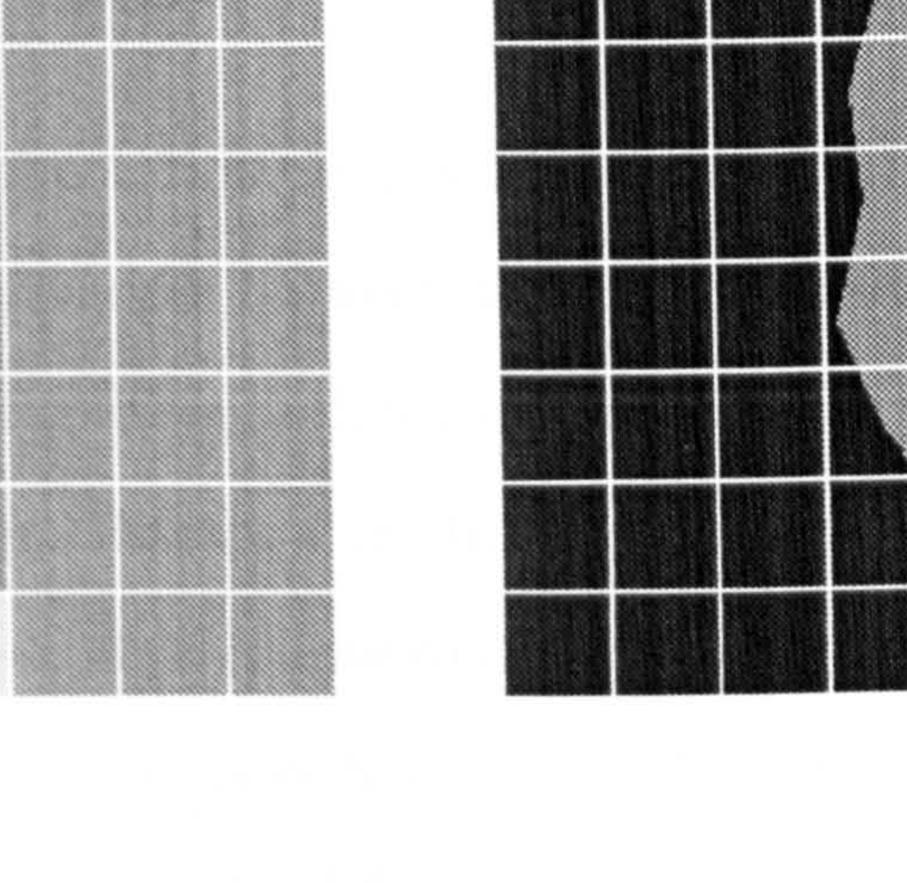
3. Calculate mean gray level of each segmented region which, in turn, is to be used at the immediate level below in the calculation of potential



(a)



(b)



(c)

Figure 5.8: Propagation of boundary detected by boundary process. a) Segmented image with boundary detected at level 3 superposed. b) Relabelling of the segmented image according to the curved boundary detected by boundary process. c) initial configuration at level 4 with the curved boundaries estimated in the boundary process at level 3 superposed for reference.

$V_f(\lambda_s, \lambda_f, X_{sf})$, σ_{wf}^2 and σ_{of}^2 in Equation (4.24).

In summary, the complete algorithm can be described as shown in Figure 5.9.

5.3 Experiments

In Chapter 4, the segmentation results of the region process were shown. To compare the performances between the algorithm utilising only region process and the one integrating both region and boundary processes, the five textured images used in Chapter 4 are tested again.

From Figure 5.10—5.14, it is easy to see that without the boundary process the estimated boundary looks blocky and the error rate is higher. This is clearest in Figure 5.10(a) at level 3 where the image is segmented into five different regions with three spurious ones along the real boundary. However, by applying the boundary process, the spurious regions are eliminated and the boundary is significantly refined as shown in Figure 5.10(b) at the same level. This not only improves the segmentation result at a given level but will also gives the next level a better initial label configuration which will result in a more accurate boundary at the next level.

The segmentation error rates of Figure 5.10—5.14 at different levels shown in Table 5.1—5.5 demonstrate the merit of the boundary process. Note that the error rates at levels 5 and 6 are similar. This represents the inherent uncertainty in the segmentation [133]. For some cases, the error rates after the execution of region process at level 6 are even higher than the error rates after the execution of boundary process at level 5. This not only represents the inherent uncertainty in the segmentation but also demonstrates the advantage of incorporating the boundary process. Table 5.1—5.5 also list the

1. high-pass filter the input image

2. current_level k = nominal_top_level

3. execute region-based process

3.1 extract texture feature measurements $X_{ss'm}$

3.1.1 perform multiresolution Fourier Transform

3.1.2 extract texture features from each sites

3.2 Segment current level using Gibbs Sampling scheme

3.2.1 for all sites, label each site based on MAP

3.2.2 if converge then continue; else go to step 3.2.1

4. execute boundary-based process

4.1 estimate boundary segments

4.1.1 estimate orientation of boundary segments

4.1.2 estimate centroid of boundary segments

4.2 identify boundary-contained blocks using Gibbs Sampling scheme

4.2.1 for all PBCB's, label each one based on MAP

4.2.2 if converge then continue; else go to step 4.2.1

4.3 connect boundary segments

5. if current_level = nominal_bottom_level then go to step 7;

else propagate information to next level (level k + 1)

6. current level k = k + 1; go to step 3

7. stop

Figure 5.9: Texture segmentation algorithm integrating region and boundary processes.

numbers of iterations per pixel ($\#\, i/p$) at different level for each experiment. Note that the $\#\, i/p$'s in the region process are copied directly from Table 4.3 because they are not expected to vary much. Also, note that $\#\, i/p$'s in the boundary process tends to increase as the algorithm descends the image pyramids because the sampling in boundary process is not conditioned on the result of previous level. Therefore, even constants C and I are given lower values at lower levels the key factor deciding $\#\, i/p$'s is the population of PBCB's. Comparing the $\#\, i/p$'s of region and boundary processes in Table 5.1—5.5, it can be seen clearly that the extra computational cost for adding boundary process is negligible.

Again, as at the end of Chapter 4, comparing Lu and co-worker's [83] result (error rate 4.5%) with ours in Table 5.4 (the classification error rate of *Image IV* in Figure 5.13), we can see that after the execution of boundary process on level 4, the error rate of our algorithm drops down to 3.3% which has shown its superiority over Lu's algorithm. At level 6, the error rate of our algorithm drops down to 2.1% only. Comparing to the best result of Wilson and Spann's algorithm applied to Figure 5.10 in [133] (1.8 classification error per boundary point or equivalent error rate 4.7%), the worst result our algorithm obtained when it is applied to *Image II* in Figure 5.11 is only at an error rate 1.9%. Actually the error rate at level 4 (2.8%) of *Image II* (see Table 5.2) — the one with the worst result, has dropped below the best result Wilson and Spann's method obtained (4.7%).

5.4 Conclusions

In this chapter, we have shown that integrating region and boundary information for segmenting textured image using multiresolution Markov Random

Table 5.1: Segmentation error rates and number of iterations per pixel (# i/p) for *Image I.*

level k	C	I	Region Process		Boundary Process	
			Error rate (%)	# i/p	Error rate (%)	# i/p
3	8	3	7.053	0.189	3.079	0.018
4	6	1	1.640	0.074	1.059	0.009
5	4	1	1.265	0.349	0.485	0.016
6	3	1	0.716	2.191	0.365	0.045
Total # i/p				2.803		0.088

Table 5.2: Segmentation error rates and number of iterations per pixel (# i/p) for *Image II.*

level k	C	I	Region Process		Boundary Process	
			Error rate (%)	# i/p	Error rate (%)	# i/p
3	8	3	13.075	0.340	7.890	0.014
4	6	1	2.945	0.129	2.779	0.017
5	4	1	2.000	0.562	1.840	0.046
6	3	1	1.851	2.626	1.877	0.100
Total # i/p				3.657		0.177

Table 5.3: Segmentation error rates and number of iterations per pixel (# i/p) for *Image III.*

level k	C	I	Region Process		Boundary Process	
			Error rate (%)	# i/p	Error rate (%)	# i/p
3	8	3	6.953	0.186	3.032	0.002
4	6	1	1.807	0.066	1.645	0.005
5	4	1	1.035	0.531	0.647	0.026
6	3	1	0.848	2.499	0.693	0.088
Total # i/p				3.282		0.121

Table 5.4: Segmentation error rates and number of iterations per pixel (# i/p) for *Image IV*.

level k	C	I	Region Process		Boundary Process	
			Error rate (%)	# i/p	Error rate (%)	# i/p
3	8	3	6.892	0.202	6.847	0.005
4	6	1	5.852	0.125	3.328	0.020
5	4	1	3.603	0.594	2.068	0.087
6	3	1	2.237	2.620	2.100	0.188
Total # i/p				3.541		0.300

Table 5.5: Segmentation error rates and number of iterations per pixel (# i/p) for *Image V*.

level k	C	I	Region Process		Boundary Process	
			Error rate (%)	# i/p	Error rate (%)	# i/p
3	8	3	2.600	0.182	4.289	0.001
4	6	1	1.645	0.058	1.816	0.007
5	4	1	1.498	0.250	1.518	0.012
6	3	1	1.395	2.064	1.170	0,029
Total # i/p				2.554		1.571

Fields (MMRF) significantly improves the segmentation results. Since the boundary process uses exactly the same statistical approach (MMRFs with simulated annealing) as that used in the region process, the generality, efficiency and consistency of the overall framework are maintained. The extra computational cost of including boundary process is insignificant relative to that of the region process because only the potential boundary-containing blocks are involved in the sampling and the calculation of interaction energy is much easier than that defined in region process.

The method currently adopted to connect the refined boundary is simply connecting the centroids of the identified neighbouring boundary blocks. This leaves room for further improvement and we are currently investigating better connecting techniques to refine the boundary.

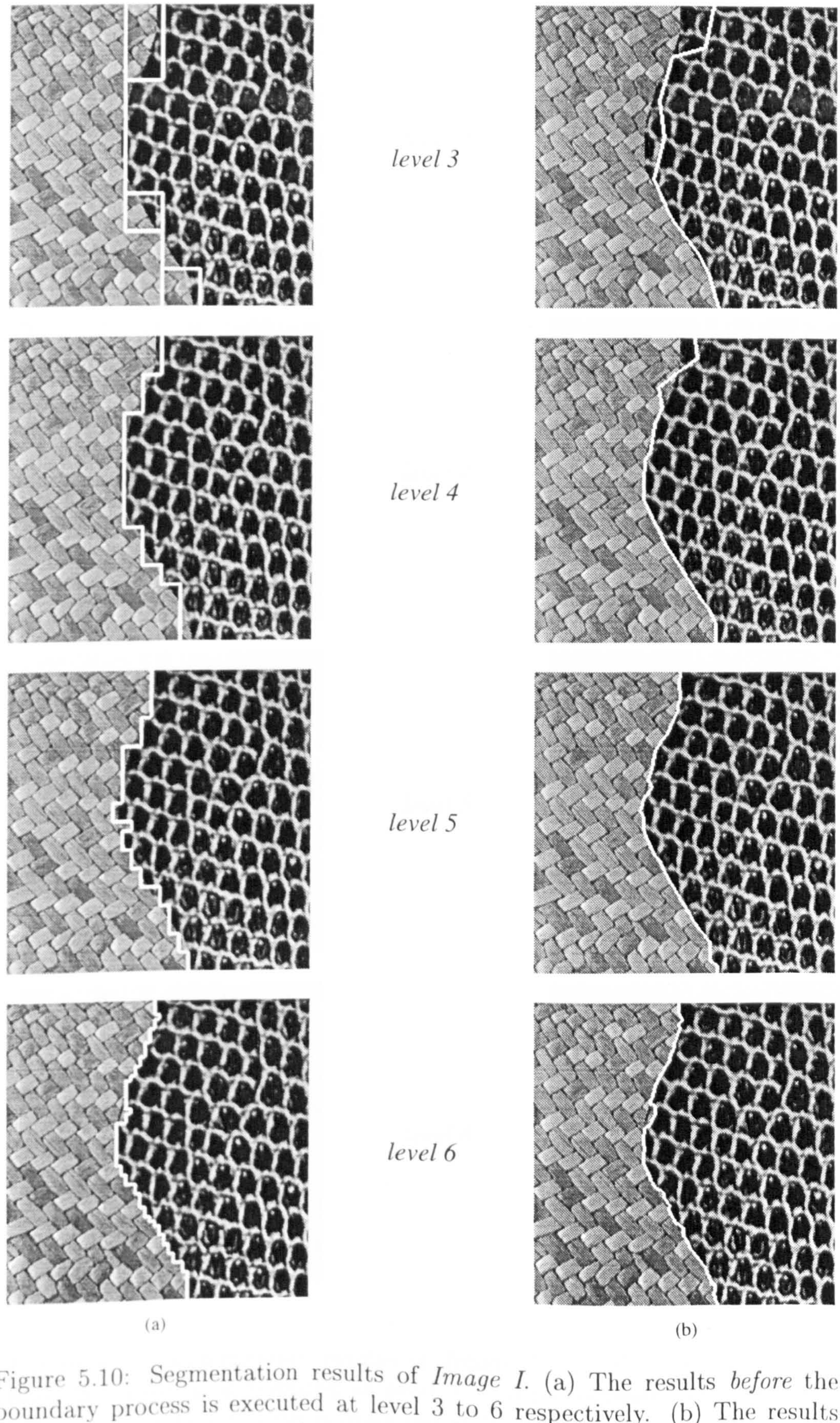


Figure 5.10: Segmentation results of *Image I*. (a) The results *before* the boundary process is executed at level 3 to 6 respectively. (b) The results *after* the boundary process is executed at level 3 to 6 respectively.

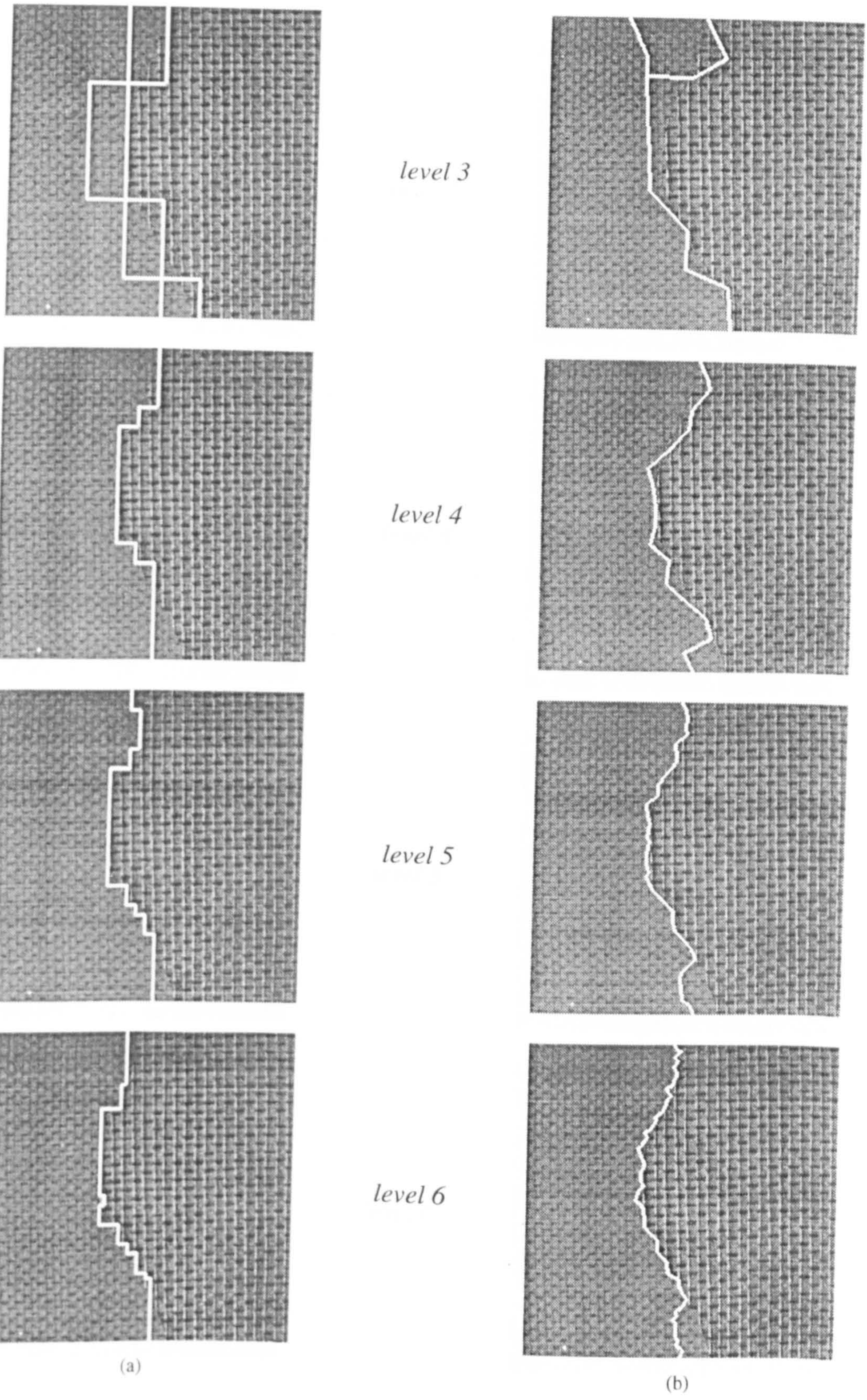


Figure 5.11: Segmentation results of *Image II*. (a) The results *before* the boundary process is executed at level 3 to 6 respectively. (b) The results *after* the boundary process is executed at level 3 to 6 respectively.

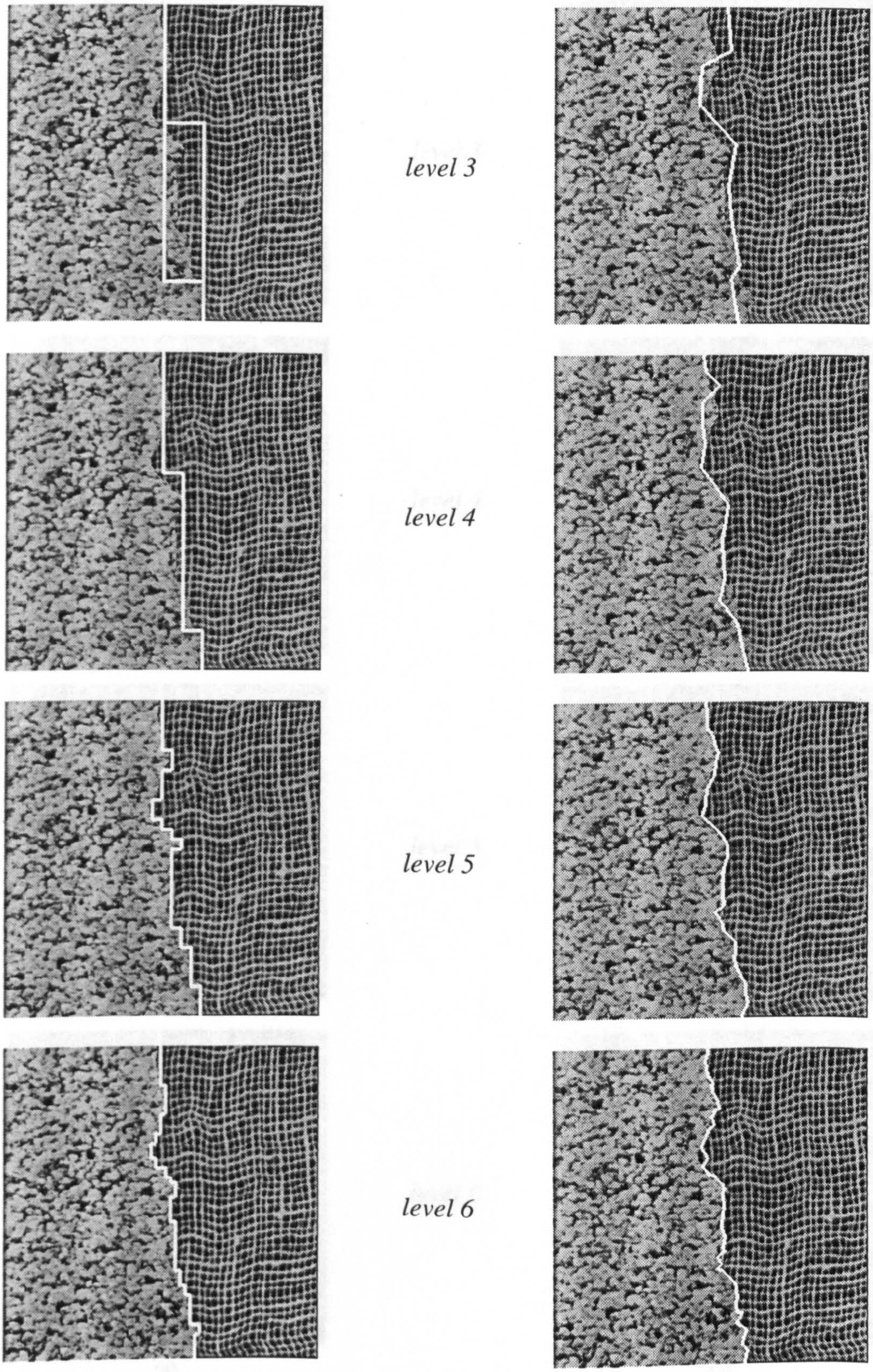


Figure 5.12: Segmentation results of *Image III*. (a) The results *before* the boundary process is executed at level 3 to 6 respectively. (b) The results *after* the boundary process is executed at level 3 to 6 respectively.

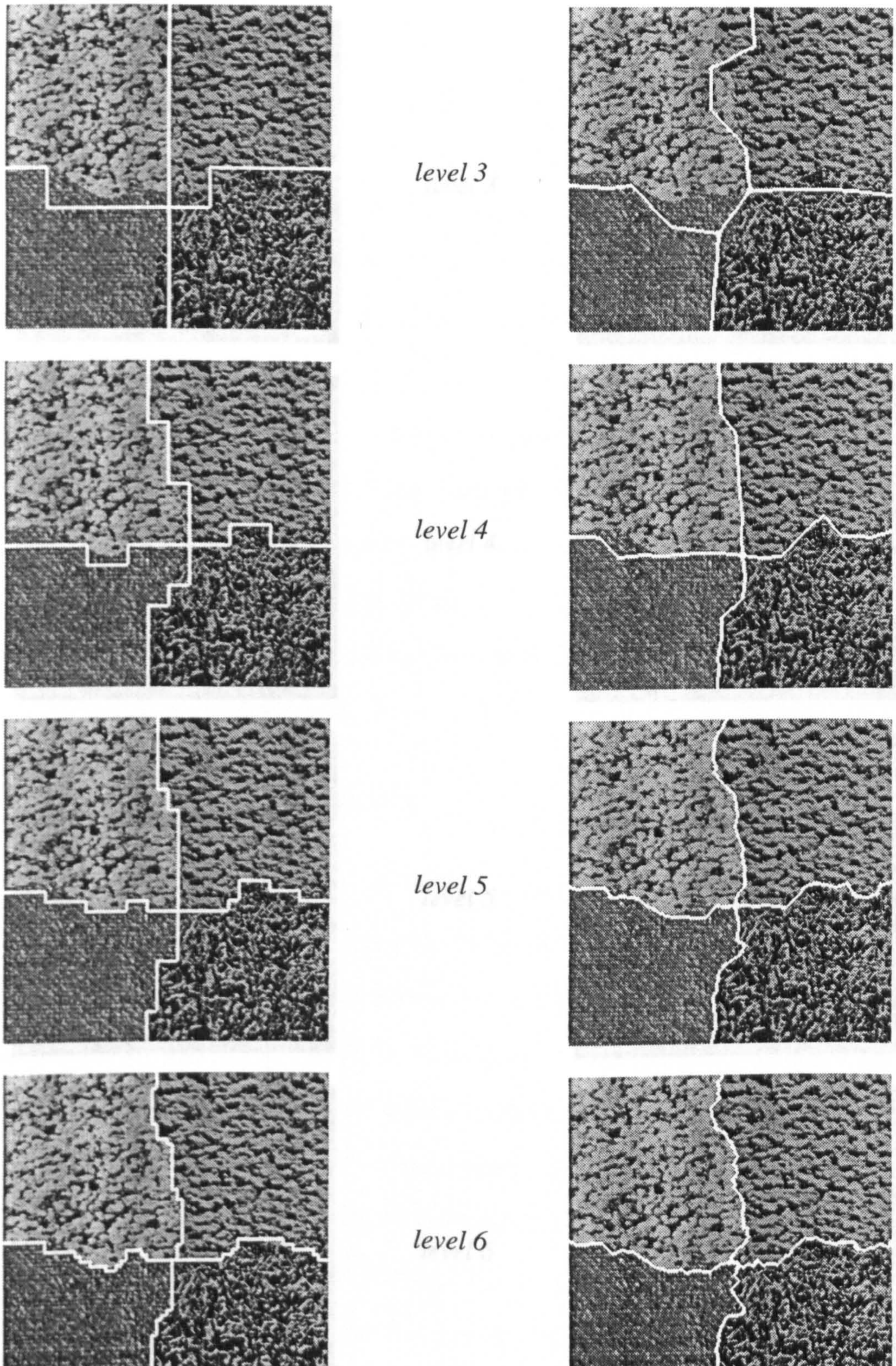
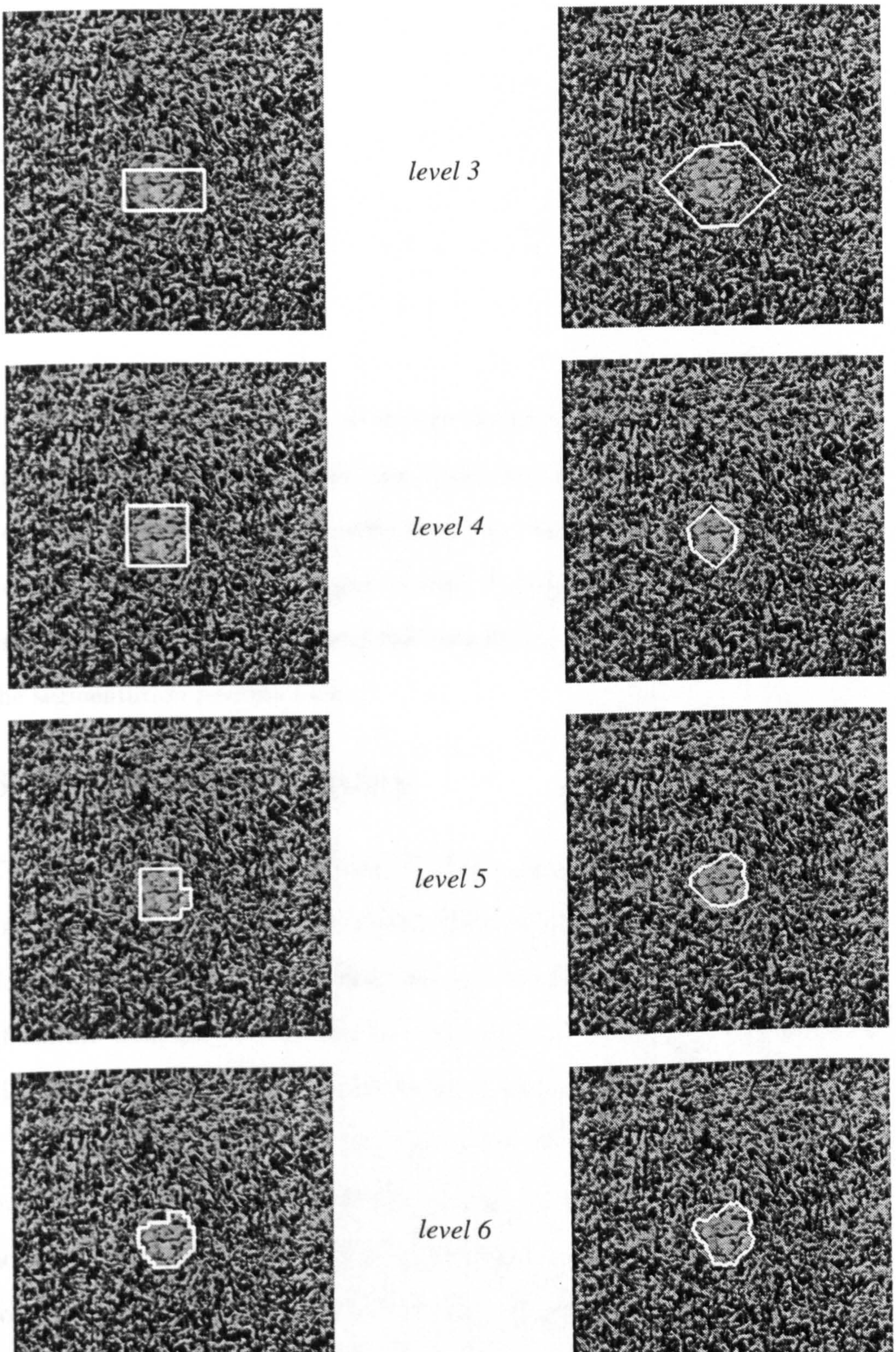


Figure 5.13: Segmentation results of *Image IV*. (a) The results *before* the boundary process is executed at level 3 to 6 respectively. (b) The results *after* the boundary process is executed at level 3 to 6 respectively.



(a) (b)

Figure 5.14: Segmentation results of *Image V*. (a) The results *before* the boundary process is executed at level 3 to 6 respectively. (b) The results *after* the boundary process is executed at level 3 to 6 respectively.

Chapter 6

Conclusion

In this thesis, a new approach to texture segmentation, based on a Multiresolution Markov Random Field, has been presented. The theory has been described and experimental results have been used to show its effectiveness in segmenting a variety of images. A way of integrating region and boundary processing was also proposed and has been shown to be effective in improving the segmentation performance.

6.1 Thesis Summary

The objectives of this thesis were to give a general idea of what texture is and what texture analysis is all about. We introduced a set of texture feature descriptors to represent the characteristics of textures and, based on this set of feature descriptors, a robust unsupervised multiresolution segmentation algorithm integrating region and boundary information.

Although the term 'texture' in the context of image processing and computer vision so far remains undefined formally, a description of the commonly accepted concept of texture is given in Chapter 1. Based on this concept, some issues frequently encountered in the literature were addressed, namely: description and extraction of features, texture discrimination and texture

segmentation.

The task of texture segmentation is non-trivial. For instance, there are no clear answers to the questions "what is an object?" and "what makes it so important that it should be treated and detected as a region from its surrounding environment?" The uncertainty about 'what is where' plays an important and tricky role in texture analysis. The majority of texture segmentation methods still perform under human supervision. These supervised segmentation methods require *a priori* knowledge about the number of texture regions present in the image and training in advance so as to establish a texture property database on which to base the discrimination and segmentation methods.

In Chapter 2, a survey of the notable methods of feature extraction and texture segmentation was carried out to reflect the issues addressed in Chapter 1 and how they have been tackled by various workers. In the latter part of Chapter 2, segmentation methods were divided into three categories: region-based, boundary-based, and hybrid categories. In general, the main difference between region-based and boundary-based methods is that the former are concerned with feature homogeneity and attempt to group pixels or blocks with similar features together, while the latter are concerned with the feature inhomogeneity and attempt to detect boundaries between blocks. Often, region processes do not make use of boundary information, so that when a relatively larger window is used, the confidence on the boundary position is limited. Furthermore, for region growing methods, segmentation results depend on the underlying growing algorithm and how the seeds are placed. For split and merge methods, the criteria for splitting and merging have to be specified. On the other hand, boundary processes seldom use

region information and as a result, ensuring boundary continuity and thinness poses a challenge to the algorithms. To overcome the shortcomings of both processes, it is preferable to integrate the two. However, how this may be possibly achieved is not obvious and so far the possibility has been less widely explored [3][77][98][138].

In Chapter 3, the goal of segmentation algorithms was discussed. Based on an understanding of the segmentation goal, we described the mathematical background of Markov Random Fields (MRF) and how multiresolution approaches can be employed to alleviate class-position uncertainty. MRF's are one of the most popular models for texture analysis in recent years. Several issues surrounding the application of MRF's were raised and some simple experimental examples which put MRF theory to the test were also given.

Despite the desirable attributes of the MRF framework, the assumption of Markovianity allows the global information such as texture features to be communicated only through local interactions within a defined neighbourhood. This inherent shortcoming calls for some remedy. In addition to this drawback, MRF models at single resolution do not provide a solution to the issue of *uncertainty* addressed in Chapter 1. Similarly, some texture features which occur at one scale or resolution do not necessarily appear at another. Applying MRF at a single resolution is likely to lose some important features which are essential to successful segmentation. To circumvent these shortcomings, multiresolution approaches appear to be intuitive and efficient with the following advantages:

- Multiple resolution approaches allows global information (e.g. texture class) and local information (e.g. texture boundary position) to propagate smoothly across different resolutions.

- Class-position (*what-where*) uncertainty is alleviated via information fusion at different resolutions, which leads to localisation in both class space and boundary position.

- Texture features occurring at different resolutions or scale can be picked up within a multiresolution scheme.

In Chapter 4, we presented a new algorithm using Multiresolution Markov Random Fields (MMRF) described in Chapter 3, which partitions a textured image into regions, with each possessing homogeneous texture properties. The algorithm consists of a texture feature extraction stage followed by a segmentation one which requires no *a priori* knowledge about the number of textured regions present in the image. The Multiresolution Fourier Transform (MFT) is first applied to a high-pass version of a textured image to create an image pyramid as described in Section 3.4. A set of four spatially localised texture features is then extracted from each block of the image pyramid to describe it. Each set of texture features is based on a two-component model of texture, in which one component is an affine deformation, representing the structural or deterministic elements and the other is a stochastic one based on the local Fourier energy spectrum. Based on the extracted feature sets, each level of the image pyramid is modelled as a Markov Random Field and stochastic relaxation is adopted to maximise the posterior likelihood in assigning one of the class labels to the block (site) being visited. Upon convergence of the algorithm at a given level, the segmentation result containing texture class and boundary information is propagated from low spatial resolution to high spatial resolution to minimise class-position uncertainty, by repeating the algorithm at the next level. Experiments on the segmentation

of natural textures at the end of the chapter have shown the computational efficiency and robustness of this multiresolution segmentation method.

However, the algorithm in Chapter 4 makes use of only region information. When the algorithm converges at each resolution level, the approximated boundary is drawn between blocks of different texture classes. This approximation inevitably results in block artifacts. Although refinement at successive levels of resolution (smaller block size) does alleviate the uncertainty, the block effect remains.

To overcome these shortcomings, a boundary-based process, using boundary information and the same MMRF framework to complement region process and give improved boundary estimates at little extra computational cost, is proposed in Chapter 5. At each resolution level, a preliminary boundary is first approximated using the region process. Following the completion of the region process, the boundary process assumes the task of boundary refinement and all the image blocks on either side of the preliminary boundary approximated are taken as potential boundary-containing blocks (PBCB's). The orientation and the centroid of the boundary-segment contained in each PBCB are estimated so that a 'distance' measure based on them can be calculated and encoded in the interaction energy function between pairs of PBCB's. The set of PBCB's is then modelled as a MRF and undergoes an optimisation process to detect the real boundary-contained blocks. Again simulated annealing is adopted to maximise the *a posteriori* probability for assigning *boundary/non-boundary* labels to the PBCB's. Upon completion of the algorithm, the refined boundary is formed by connecting the centroids of the blocks with *boundary* label and propagated to the next resolution for further refinement.

6.2 Limitations and Future Work

Although the complementary boundary process does improve the segmentation results of the region process, the two processes are performed sequentially in this work, i.e. the regional information and boundary information are not employed concurrently. Should there be any 'leakages' – small disparities between neighbouring blocks belonging to different regions, across the texture boundary, the region process will have difficulty detecting the boundary and sites on both sides of the boundary will be classified as belonging to the same class. The boundary process is not designed to counter the mislabelling due to leakages, but only to identify the boundary-containing blocks along the preliminary boundary estimated by the region process. Thus, this leakage effect can result in having two or more regions with significantly different textural properties being assigned the same class labels and being connected with narrow 'bridges' over the leakages. The whole algorithm can thus be defeated by tiny leakages. One way to prevent this from happening is to make sure that between any site pairs, at least one of the components of the textural feature set is sensitive enough to detect the presence of a boundary, should there be one. This reasoning seems to suggest the use of a bigger feature set. However, there is still no guarantee that leakages can be prevented and on the other hand, as we mentioned in Chapter 2, too many features will downgrade the efficiency of the segmentation algorithm and the segmentation result. We expect that if the region and boundary processes are executed concurrently then the problem of leakage can be tackled without using a larger feature set.

The method we proposed in Section 4.2.3 to counter the deadlock (some-

times called over-segmentation) problem is by and large heuristic. Since there is no real boundary or significant difference between over-segmented neighbouring regions, executing region and boundary processes concurrently may be able to avoid this problem, because boundary-containing blocks only emerge along real boundary.

The method currently adopted in boundary process to connect the refined boundary is simply connecting the centroids of the identified neighbouring boundary blocks. This leaves room for further improvement and we are currently investigating connecting techniques to refine the boundary.

6.3 Concluding Remarks

In summary, the main contributions of this work are:

1. Multiresolution segmentation is placed on a sound footing using MRF's in conjunction with stochastic relaxation.

2. A way of combining region and boundary processing to improve the performance is found.

3. The two-component texture model proposed by Hsu [49] is shown to be useful in segmentation. We demonstrate that the warping error is a good measure to discriminate structural textures.

4. The algorithm is designed to work without human supervision. *a priori* knowledge on the features and number of textures in the input image need not to be specified, so a training phase is unnecessary.

Although much work remains to be done, the results obtained so far compare favourably with those reported elsewhere.

Appendix A

List of Publications

- Chang-Tsun Li, Roland Wilson, "Textured Image Segmentation Using Multiresolution Markov Random Fields and a Two-component Texture Model", in Proc. 10th Scandinavian Conference on Image Analysis, Lappeenranta, Finland, 1997.

- Chang-Tsun Li, Roland Wilson, "MRF Approach to Texture Segmentation Integrating Region and Boundary Information", in Proc. IASTED International Conference on Signal and Image Processing, New Orleans, USA, 1997.

- Chang-Tsun Li, Roland Wilson, "Image Segmentation Based on a Multiresolution Bayesian Framework", Submitted to IEEE International Conference on Image Processing, 1998.

Bibliography

[1] *Collins English Dictionary.* HarperCollins Publishers, 1994.

[2] F. Ade. Characterisation of Textures by eigenfilters. *Signal Processing*, 5:451–457, 1983.

[3] N. Ahuja. A Transform for Multi-scale Image Segmentation by Integrated Edge and Region Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(12):1211–1235, 1996.

[4] Robert Azencott, Jia-Ping Wang, and Laurent Younes. Texture classification using windowed fourier filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(2), 1997.

[5] J. Beck. Textural Segmentation. In *Organisation and Representation in Perception, ed. by J. Beck, Hillsdale, Lawrence Erlbaum Associates, 285-317*, 1982.

[6] J. Besag. Spatial Interaction and the statistical Analysis of Lattice Systems. *Journal of the Royal Statistical Society (Series B)*, 36:192–236, 1974.

[7] A. Bhalerao and R. Wilson. Multiresolution Image Segmentation Combining Region and Boundary Information. In *Theory and Applications of Image Analysis*, 1992.

[8] J. Bigun and J. M. Du Buf. N-Folded Symmetries by Complex Moments in Gabor Space and Their Application to Unsupervised Texture Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(1):80–87, 1994.

[9] G. Boccignone, M. G. Mancini, and A. Marcelli. Solving Diagnostic Analysis of Ancient Buildings Degradation as a Texture Discriminant Analysis Problem. In *Proc. 7th Scandinavian Conference on Image Analysis, Aalborg, Denmark*, 1991.

[10] A. Bovik, M. Clark, and Geisler. Multichannel Texture Analysis Using Localized Spatial Filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(1):55–72, 1990.

[11] C. Brice and C. Fennema. Scene Analysis using regions. *Artificial Imtelligence*, 1:205–226, 1970.

[12] P. Brodatz. *Textures-A Photographic Album for Artists and Designers.* Dover, New York, 1966.

[13] P. J. Burt and E. H. Adelson. The Laplacian Pyramid as a Compact Image Code. *IEEE Transactions on Communication*, COM-31:532–540, 1983.

[14] T. Caelli. On Discriminating Visual Textures and Images. *Perception and Psychophysics*, 31:149–159, 1982.

[15] A. Calway. *The Multiresolution Fourier Transform: A Genera Purpose Tool for Image Analysis.* PhD thesis, Department of Computer Science, The University of Warwick, UK, September 1989.

[16] J. M. Carstensen. Co-occurrence Feature Performance in Texture Classification. In *Proc. 8th Scandinavian Conference on Image Analysis, Tromso, Norway*, 1993.

[17] R. Chellappa and r. L. Kashyap. Digital Image Restoration Using Spatial Interaction Models. *IEEE Transactions on Acoustics, Speech, Signal Processing*, 30:461–472, 1982.

[18] R. Chellappa and r. L. Kashyap. Texture Synthesis using 2-D noncausal autoregressive models. *IEEE Transactions on Acoustics, Speech, Signal Processing*, 33:194–203, 1985.

[19] C. H. Chen. *Nonlinear Maximum Entropy Spectral Analysis Methods for Signal Recognition.* Research Studies Press, Chichester, 1982.

[20] J. L. Chen and A. Kundu. Rotation and Gray-Scale Transform Invariant Texture Identification using Wavelet Decomposition and Hidden Markov Model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(2):208–214, 1994.

[21] J. L. Chen and A. Kundu. Unsupervised Texture Segmentation using Multichannel Decomposition and Hidden Markov-Models. *IEEE Transactions on Image Processing*, 4(5):603–619, 1995.

[22] P. C. Chen and T. Pavlidis. Image Segmentation as Estimation. In *Image Modeling*, , ed. by A. Rosenfeld, Academic Press, 1980.

[23] Y. Chen and E. R. Dougherty. Gray-scale Morphological Granulometric Texture Classification. *Optical Engineering*, 33(8):2713–2722, 1994.

[24] Y. Q. Chen, M. S. Nixon, and D. W. Thomas. Statistical Geometrical Features for Texture Classification. *Pattern Recognition*, 28:537–552, 1995.

[25] F. S. Cohen and Z. Fan. Maximum Likelihood Unsupervised Textured Image Segmentation. *Computer Vision, Graphics, and Image Processing*, 54:239–251, 1992.

[26] R. W. Conners and C. A. Harlow. A Theoretical Comparison of Texture Algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2(3):204–222, 1980.

[27] G. R. Cross and A. K. Jain. Markov Random Field Texture Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5:25–39, 1983.

[28] F. D'Astous and M. E. Jernigan. Texture Discrimination Based on Detailed Measures of the Power Spectrum. In *Proc. 7th International Conference on Pattern Recognition, Montreal, Canada*, 1984.

[29] Andrew R. Davies. *Image Feature Analysis using the Multiresolution Fourier Transform*. PhD thesis, Department of Computer Science, The University of Warwick, UK, 1993.

[30] I. Dinstein, A. C. Fong, L. M. Ni, and K. Y. Wong. Fast Discrimination between Homogeneous and Textured Regions. In *Proc. 7th International Conference on Pattern Recognition, pp. 361-363, Montreal, Canada,*, 1984.

[31] E. R. Dougherty, J. T. Newell, and J. B. Pelz. Morphological Texture-based Maximum-Likelihood Pixel Classification Based on Local Granulometric Moment. *Pattern Recognition*, 25:1181–1198, 1992.

[32] D. Dunn and W.E. Higgins. Optimal Gabor Filters for Texture Segmentation. *IEEE Transactions on Image Processing*, 4(7):947–964, 1995.

[33] D. Dunn, W.E. Higgins, and J. Wakeley. Texture Segmentation using 2-D Gabor Elementary-Functions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(2):130–149, 1994.

[34] R. W. Ehrich and J. P. Foith. Representation of Random Waveforms by Relation Trees. *IEEE Transactions on Computers*, C-25:725–735, 1976.

[35] D. Gabor. Theory of Communication. *J. Inst Elect. Engr.*, 93:429–455, 1946.

[36] D. Geman, C. Graffigne S. Geman, and P Dong. Boundary Detection by Constrained Optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12:609–628, 1990.

[37] S. Geman and D. Geman. Stochastic relaxation, Gibbs distribution, and Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741, 1984.

[38] Nathalie Giordana and Wojciech Pieczynski. Estimation of Generalised Multi-sensor Hidden Markov Chains and Unsupervised Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(5):465–475, 1997.

[39] R. C. Gonzalez and R. E. Woods. *Digital Image Processing*. Addison Wesley, 1992.

[40] L. V. Gool, P. Dewaele, and A. Oosterlinck. Survey: Texture Analysis Anno 1983. *Computer Vision, Graphics, and Image Processing*, 29:336–357, 1985.

[41] F. Goudail, E Lange, and N. Otsu K. Kyuma. Face recognition system using local autocorrelations and multiscale integration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(10):1024–1028, 1996.

[42] G. H. Granlund and H. Knutsson. *Signal Processing for Computer vision, Ch. 13.* Kluwer Academic Publishers, 1995.

[43] J. F. Haddon and J. F. Boyce. Image Segmentation by Unifying Region and Boundary Information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12:929–948, 1990.

[44] R. Haralick, K. Shanmugam, and H. Dinstein. Texture Features for Image Classification. *IEEE Transactions on Systems, Man, and Cybernetics*, 3(6):610–621, 1973.

[45] R. M. Haralick. Statistical and Structural Approaches to Texture. *Proceedings of the IEEE*, 67(5):786–804, 1979.

[46] R. M. Haralick. Image Texture Survey. In *Fundamentals in Computer Vision*, ed. by O. D. Faugeras, Cambridge University Press, 1983.

[47] S. L. Horowitz and T. Pavlidis. Picture Segmentation by a Tree Traversal algorithm. *T. Assoc. Computer. Mach.*, 23:368–388, 1976.

[48] W. Hsiao and A. A. Sawchuk. Unsupervised Textured Image Segmentation using Feature Smoothing and Probabilistic Relaxation Techniques. *Computer Vision, Graphics, and Image Processing*, 48:1–21, 1989.

[49] T. I. Hsu. *Texture Analysis and Synthesis using Multiresolution Fourier Transform.* PhD thesis, Department of Computer Science, The University of Warwick, UK, 1994.

[50] T. I. Hsu and R. G. Wilson. A Two-component Model of Texture for Analysis and Synthesis. *IEEE Transactions on Image Processing*, to be published, 1998.

[51] Tao-I Hsu, A.D. Calway, and R. Wilson. Texture Analysis using the Multiresolution Fourier Transform. In *Proc. 8th Scandinavian Conference on Image Analysis*, 1993.

[52] Tao-I Hsu and R. Wilson. Texture Analysis using Generalised Wavelet Transform. In *Proc. ICIP '94, Austin, Texas, U.S.A.*, 1994.

[53] F. Huet and T. Mattioli. A Textural Analysis by Mathematical Morphology. In *Proc. International Symposium on Mathematical Morphology*, Atlanta, USA, 1996.

[54] F. Huet and T. Mattioli. A Textural Analysis Mathematical Morphology Transformations: Structural Opening and Top-Hat. In *Proc. IEEE International Conference on Image Processing (ICIP96)*, Lausanne, Switzerland, 1996.

[55] A. Jain and F. Farrokhnia. Unsupervised Texture Segmentation using Gabor Filters. *Pattern Recognition*, 24(12):1167–1186, 1991.

[56] X. G. Jia and M. S. Nixon. Extending the Feature Vector for Automatic Face Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(11):1167–1176, 1995.

[57] B. Julesz. Visual Pattern Discrimination. *IRE Trans. Inform. Theory*, 8, No. 2:84–92, 1962.

[58] B. Julesz. Perceptual Limits of Texture Discrimination and their Implications to Figure-Ground Separation. In *Formal Theories of Visual Textures Perception, ed. by E. L. J. Leeuwenberg, H. F. J. M. Buffart, Chichester, Wiley*, 1978.

[59] B. Julesz. Textons, the Elements of Texture perception, and their Interactions. volume 290, March, 1981.

[60] H. Kaizer. A Quantification of textures on Aerial Photographs. Technical Report Tech. Note 121, Boston University Research Laboratories, Boston University, Boston, MA. USA, 1955.

[61] L. M. Kaplan and C. Kuo. Roughness Analysis and Synthesis via Extended Self-Similar (ESS) Model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17:1043–1056, 1995.

[62] B. Kartikeyan and A. Sarkar. A Identification Approach for 2-D AR Models for Describing Textures. *CVGIP: Graphics, Models, Image Processing*, 53:121–131, 1991.

[63] K. Karu, A. K. Jain, and R. M. Bolle. Is There any Texture in the Image? *Pattern Recognition*, 29(9):1437–1446, 1996.

[64] R. L. Kashyap. Characterization and Estimation of Two Dimensional ARMA Models. *IEEE Transactions on Information Theory*, 30:736–745, 1984.

[65] R. L. Kashyap and R. Chellappa. Estimation and Choice of Neighbours in Spatial Interaction Models of Images. *IEEE Transactions on Information Theory*, 29:60–72, 1983.

[66] J. Keller, R. Crownover, and S. Chen. Texture Description and Segmentation Through Fractal Geometry. *Computer Vision, Graphics, and Image Processing*, 45:150–160, 1989.

[67] C. Kervrann and F. Heitz. A Markov Random Field Model Based Approach To Unsupervised Texture Segmentation Using Local And Global Spatial Statistics. *IEEE Transactions on Image Processing*, 4, 1995.

[68] A. Khotanzad and J. Y. Chen. Unsupervised Segmentation of Textured Images by Edge Detection in Multidimensional Features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(4):414–421, 1989.

[69] R. Kinderman and J. L. Snell. *Markov Random Fields and Their Applications* . American Math. Society, 1980.

[70] H. Knutsson and G. H. Granlund. Texture Analysis Using Two-dimensional Quadrature Filters. In *Proc. IEEE Computer Society Workshop on Computer Architecture for Pattern Analysis and Image Database Management, Pasadena, CA, USA*, 1983.

[71] S. Krishnamachari and R. Chellappa. Multiresolution Gauss-Markov Random Field Models for Texture Segmentation. *IEEE Transactions on Image Processing*, 6(2):251–267, 1997.

[72] P. Kube and A. Pentland. On the Imaging of Fractal Surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10:704–707, 1988.

[73] A. Laine and J. Fan. Frame Representations for Texture Segmentation. *IEEE Transactions on Image Processing*, 5(5):771–780, 1996.

[74] K. I. Laws. *Textured Image Segmentation.* PhD thesis, University of Southern California, USA, 1980.

[75] T.S. Lee. Image representation using 2d gabor wavelets. *PAMI*, 18(10):959–971, 1996.

[76] C. T. Li and R. Wilson. Image Segmentation Using Multiresolution Fourier Transform. Technical report, Department of Computer Science, University of Warwick, 1995.

[77] C. T. Li and R. Wilson. MRF Approach to Texture Segmentation Integrating Region and Boundary Information. In *Proc. IASTED International Conference on Signal and Image Processing, New Orleans, USA*, Dec 1997.

[78] S. Z. Li. *Markov Random Field Modeling in Computer Vision.* Springer-Verlag, 1995.

[79] W. Li, V. Haese-Coat, and J. Ronsin. Residues of Morphological Filtering by Reconstruction for Texture Classification. *Pattern Recognition*, 30(7):1081–1093, 1997.

[80] Kung-Hao Liang. *From Uncertainty to Adaptivity: Multisacle Edge Detection and Image Segmentation.* PhD thesis, University of Warwick, UK, 1997.

[81] F. Liu and R. W. Picard. Periodicity, Directionality, and Randomness: Wold Features for Image Modelling and Retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(7):722–733, 1996.

[82] S. S. Liu and M. E. Jernigan. Texture Analysis and Discrimination in Additive Noise. *Computer Vision, Graphics, and Image Processing*, 49:52–67, 1990.

[83] C. S. Lu, P. C. Chung, and C. F. Chen. Unsupervised Texture Segmentation via wavelet Transform. *PR*, 30(5):729–742, 1997.

[84] D. Maio and D Maltoni. Direct gray-scale minutiae detection in fingerprints. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(1):27–40, 1997.

[85] B. B. Mandelbrot. *Fractal Geometry of Nature.* Freeman, 1982.

[86] B. S. Manjunath and R. Chellappa. Unsupervised Texture Segmentation Using Markov Random Field Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13:478–482, 1991.

[87] B. S. Manjunath and M. A. Wy. Texture Features for Browsing and Retrieval of Image Data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18:837–824, 1996.

[88] J. Mao and A. K. Jain. Texture Classification and Segmentation Using Multiresolution Simultaneous Autoregressive Models. *Pattern Recognition*, 25(2):173–188, 1992.

[89] David Marr. *Vision.* Freeman, 1982.

[90] T. Matsuyama, S. Miura, and M. Nagao. A Structural Analysis of Natural Textures by Fourier Transformation. In *Proc. 6th Internal Conference on Pattern Recognition, Munich, Germany*, pages 289–292, 1982.

[91] T. Matsuyama, K. Saburi, and M. Nagao. A Structural Analyzer for Regularly Arranged Textures. *Computer Graphics and Image Processing*, 18:259–278, 1982.

[92] L. O'gorman and A. C. Sanderson. The converging squares algorithm: an efficient method for locating peaks in multidimensions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:280–288, 1984.

[93] P. P. Ohanian and R. C. Dubes. Performance Evaluation for 4 Classes of Textural Features. *Pattern Recognition*, 25:819–833, 1992.

[94] R. Ohlander. *Analysis of Natural Scenes.* PhD thesis, Carnegie-Mellon University, PA, USA, 1975.

[95] Athanasios Papoulis. *Signal Analysis.* McGraw Hill, 1981.

[96] D. Patel, E. R. Davies, and I. Hannah. The Use of Convolution Operations for Detecting Contaminant in Food Images. *Pattern Recognition*, 29(6):1019–1029, 1996.

[97] D. Patel and T. J. Stonham. Texture Based Image Segmentation. In *Proc. 7th Scandinavian Conference on Image Analysis, Aalborg, Denmark*, 1991.

[98] T. Pavlidis and Y. T. Liow. Integrating Region Growing and Edge Detection, Technical Report TR.89.01.05. Technical report, Department of Computer Science, State University of New York, Stony Brook, 1989.

[99] A. P. Pentland. Fractal Based Description of Natural Scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:661–674, 1984.

[100] M. Pietikäinen, A. Rosenfeld, and L. S. Davis. Experiments with Texture Classification using Averages of Local Pattern Matches. *IEEE Transactions on Systems, Man, and Cybernetics*, 13:421–426, 1983.

[101] J. Puzicha, T. Hofmann, and J. M. Buhmann. Non-Parametric Similarity Measures for Unsupervised Texture Segmentation and Image Retrieval. In *Proc. IEEE Intl. Conf. on Computer Vision and Pattern Recognition, Puerto Rico*, 1997.

[102] T. Randen and J. H. Husoy. Multichannel filtering for Image Texture Segmentation. *Opt. Eng.*, 33(8):2617–2625, 1994.

[103] A. R. Rao and R. C. Jain. Computerized Flow Field Analysis - Oriented Texture Fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(7):693–709, 1992.

[104] G. Ravichandran and M. M. Trivedi. Circular-Mellin Features for Texture Segmentation. *IEEE Transactions on Image Processing*, 4(12):1629–1640, 1995.

[105] T. R. Reed and J. M. Hans Du Buf. A Review of Recent Texture Segmentation and Feature Extraction Techniques. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 57:359–372, 1993.

[106] A. Rosenfeld and A. C. Kak. *Digital Picture Processing.* Academic, 1982.

[107] A. Rosenfeld and E. B. Troy. Visual Texture Analysis. In *Proc. Symposium on Feature Extraction and Selection in Pattern Recognition*, 1970.

[108] A. Sarkar, K. M. S. Sharma, and R. V. Sonak. A New Approach for Subset 2-D AR Model Identification for Describing Textures. *IEEE Transactions on Image Processing*, 6(3):407–413, 1997.

[109] N. Sarkar and B. B. Chaudhuri. A Efficient Approach to Estimate Fractal Dimension of Textural Image. *Pattern Recognition*, 25(9):1035–1042, 1992.

[110] N. Sarkar and B. B. Chaudhuri. Multifractal and Generalized Dimensions of Gray-Tone Digital Images. *Signal Processing*, 42:181–190, 1995.

[111] P. De Souza. Texture Recognition via Autoregression. *Pattern Recognition*, 15:471–475, 1982.

[112] M. Spann and R. Wilson. A Quad-Tree Approach to Image Segmentation Which Combines Statistical and Spatial Information. *Pattern Recognition*, 18(3/4), 1985.

[113] Michael Spann. *Texture Description and Segmentation in Image Processing.* PhD thesis, The University of Aston in Birmingham, UK, 1985.

[114] B. J. Super and A. C. Bovik. Shape from Texture using Local Spectral Moments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(4):333–343, 1995.

[115] G. Székely, A. Kelemen, C. Brechbühler, and G. Gerig. Segmentation of 2-D and 3-D objects from MRI volume data using Constrained Elastic Deformations of Flexible Fourier Contour and Surface Models. *Medical Image Analysis*, 1(1):19–34, 1996.

[116] H. Tamura. Texture Features Corresponding to Visual Perception. *IEEE Transactions on Systems, Man, and Cybernetics*, 8:460–472, 1978.

[117] T. N. Tan. Texture Edge Detection by Modelling Visual Cortical Channels. *Pattern Recognition*, 28(9):1283–1298, 1995.

[118] A. Teuner, O. Picher, and B. J. Hosticka. Unsupervised Texture Segmentation of Images using Tuned Matched Gabor Filters. *IEEE Transactions on Image Processing*, 4(6):863–870, 1995.

[119] F. Tomita. Hierarchical Description of Textures. In *Proc. IJCAI-81*, pages 728–733, 1981.

[120] F. Tomita, Y. Shirai, and S. Tsuji. Description of Textures by a Structural Analysis. In *Proc. IJCAI-79*, pages 564–571, 1979.

[121] M. Unser. Local Linear Transforms for Texture Measurements. *Signal Processing*, 11:61–79, 1986.

[122] M. Unser. Texture Classification and Segmentation using Wavelet Frames. *IEEE Transactions on Image Processing*, 4:1549–1560, 1995.

[123] R. L. De Valois and K. K. De Valois. *Spatial Sision.* Oxford University Press, Oxford, 1988.

[124] D. Wang. Texture Features Based on Mathematical Morphology and Their Efficiency Comparison. In *Proc. International Conference on Signal Processing, Beijing, China*, 1993.

[125] D. Wang, V. Haese-Coat, A. Bruno, and J. Ronsin. Texture Classification and Segmentation Based on Iterative Morphological Decomposition. *J. Visual Comm. Image Representation*, 4:197–214, 1993.

[126] R. Wang, A. R. Hanson, and E. M. Riseman. Texture Analysis based on Local standard Deviation of Intensity. In *Proc. IEEE International Conference on Computer Vision and Pattern Recognition, Miami Beach*, 1986.

[127] T. P. Weldon, W. E. Higgins, and D. F. Dunn. Efficient Gabor Filter Design for Texture Segmentation. *Pattern Recognition*, 29(12):2005–2015, 1996.

[128] D. Wermser. Unsupervised Segmentation by Use of a Texture Gradient. In *Proc. 7th Intl. Conf. on Pattern Recognition, Montreal, Canada*, 1984.

[129] Carl-Fredrik Westin. *A Tensor Framework for Multidimensional Signal Processing.* PhD thesis, Department of Electrical Engineering, Linköping University, Sweden, 1994.

[130] J. S. Weszka, C. Dyer, and A. Rosenfeld. A Comparative Study of Texture Measures for Terrain Classification. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-6:269–285, 1976.

[131] R. Wilson and A. H. Bhalerao. Kernel Designs for Efficient Multiresolution Edge Detection and Orientation Estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14:384–390, 1992.

[132] R. Wilson, A. Calway, and E.R.S. Pearson. A Generalised Wavelet Transform for Fourier Analysis: The Multiresolution Fourier Transform and Its Application to Image and Audio Signal Analysis. *IEEE Transactions on Information Theory*, 38(2), March 1992.

[133] R. Wilson and Michael Spann. *Image Segmentation and Uncertainty.* Research Studies Press, 1988.

[134] C. H. Wu and P. C. Doerschuk. Tree Approximations to Markov Random-Fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(4), 1995.

[135] C. M. Wu and Y. C. Chen. Statistical Feature Matrix for Texture Analysis. *CVGIP: Graphics, Models, Image Processing*, 54:407–419, 1992.

[136] G. Y. Xu and K. S. Fu. Natural Scene Segmentation Based on Multiple threshold and Texture Measurement. In *Proc. 7th International Conference on Pattern Recognition, Montreal, Canada*, 1984.

[137] R. Zarita and S. Lelandais. Wavelets and High Order Statistics for Texture Classification. In *Proc. 10th Scandinavian Conference on Image Analysis, Lappeenranta, Finland*, 1997.

[138] S. C. Zhu and A. Yuille. Region Competition: Unifying Snakes, Region Growing, and Bayes/MDL for Multiband Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(9):884–900, 1996.

[139] S. W. Zucker. Multiple-Level Representations for Texture Discrimination. In *Proc. Intl. Conf. on Pattern Recognition and Image Processing, Dallas, Texas, USA*, pages 609–614, 1981.

[140] S. W. Zucker and D. Terzopoulous. Finding Structure in Co-occurrence Matrices for Texture Analysis. *CGIP*, 12:286–308, 1980.